**ROMÂNIA**

**MINISTERUL APĂRĂRII NAȚIONALE**

**ACADEMIA TEHNICĂ MILITARĂ „FERDINAND I”**

**FACULTATEA DE COMUNICAȚII ȘI SISTEME ELECTRONICE**

**PENTRU APĂRARE ȘI SECURITATE**

**Specializarea: Comunicații pentru apărare și securitate**



**PROIECT DE DIPLOMĂ**

**Identificarea și clasarea semnalului acustic emis prin dactilografiere**

**Identification and classification of the acoustic signal emitted through typing**

CONDUCĂTOR ȘTIINȚIFIC:

**Col. Conf. Univ. Dr. Ing. Florin POPESCU**

ABSOLVENT:

**Std. Sg. Maj. Isabel-Elena Costrachevici**

Conține file

Inventariat sub nr.

Poziția din indicator:

Termen de păstrare:

Referat (1/2)

Referat (2/2)

DECLARAȚIE PR PROPRIE RĂSPUNDERE PRIVIND ORIGINALITATEA CONȚINUTULUI LUCĂRII DE LICENȚĂ

Subsemnata Costrachevici Isabel-Elena domiciliată în Mun. Piatra Neamț, str. Mihai Viteazu, nr. 27, bl. C5, sc. E, Jud. Neamț, posesor al C.I. seria NZ, nr. 107196, CNP 6020809270011, eliberat de SPCLEP Piatra Neamț, autorul lucrării de licență cu titlul „Identificarea și clasarea semnalelor audio emise prin dactilografiere”, elaborată în vederea susținerii examenului de finalizare a studiilor în sesiunea iulie 2025, declar pe propria răspundere că această lucrare este rezultatul propriei activități intelectuale, nu conține porțiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislației române și a convențiilor internaționale privind drepturile de autor.

Data Semnătura

Detaliere (1/2)

Detaliere (2/2)

**Abstract**

This project is aimed at developing a system capable of recognizing and classifying the acoustic signals emitted during keyboard typing with the goal of finding precise differences between the individual keystrokes based on their acoustic characteristics. The typing acoustics can potentially give away a lot of information, which is why it is important to study and classify these signals in a security and authentication context. The research was done step-by-step and is structured around several key points.

The first step is represented by an analysis of the aspects involved, such as studying techniques of audio feature extraction and learning about deep learning algorithms.

The next part involves preprocessing the dataset proposing and testing different architectures designed to complete the main task. To extract the acoustic characteristics, the sounds are turned into visual representations using different techniques such as Melspectrograms, Mel-Frequency-Cepstral-Coeffiecients and Continuous Wavelet Transform. The proposed deep learning architectures are: a basic Convolutional Neural Network and a modified version of the CoAtNet algorithm. The models were trained using a custom dataset and different architectures in order to compare different scenarios.

In order to see actual results, the last step is meant to generate predictions using the trained models. The sequence of characters obtained could contain errors and to address this, Natural Language Processing techniques are applied to improve the word’s correctness.

**Rezumat**

Acest proiect propune dezvoltarea unui sistem capabil să recunoască și să clasifice semnalele acustice emise în timpul tastării, cu scopul de a identifica diferențele dintre tastele individuale, pe baza caracteristicilor lor acustice. Sunetele tastelor pot dezvălui o cantitate semnificativă de informații, de aceea este important ca aceste semnale să fie studiate și clasificate în contextul securității și autentificării. Cercetarea a fost realizată pas cu pas și este structurată în jurul unor puncte cheie.

Primul pas este reprezentat de o analiză a aspectelor implicate, precum studiul tehnicilor de extragere a caracteristicilor audio și înțelegerea funcționării algoritmilor de învățare profundă.

Următoarea etapă presupune preprocesarea setului de date, propunerea și testarea diferitelor arhitecturi concepute pentru a îndeplini sarcina principală a proiectului. Pentru a extrage caracteristicile acustice, sunetele au fost transformate în reprezentări vizuale, folosind tehnici cum ar fi: Melspectrograme, Coeficienții Mel-cepstrali și Transformata Wavelet Continuă. Arhitecturile propuse sunt: o rețea neuronală convoluțională clasică și o versiune modificată a algoritmului CoAtNet. Modelele au fost antrenate folosind un set de date propriu și diverse arhitecturi, pentru a realiza comparații între diferite scenarii.

Pentru a putea observa un scenariu mai realist, ultima etapă este reprezentată de generarea predicțiilor, utilizând modelele antrenate. Secvențele de caractere obținute pot conține erori, iar pentru a soluționa acest aspect, sunt aplicate tehnici de Procesare a Limbajului Natural pentru a îmbunătăți corectitudinea cuvintelor.

**CUPRINS**

[Listă de figuri 12](#_Toc197604423)

[Listă de tabele 13](#_Toc197604424)

[Listă de abrevieri 15](#_Toc197604425)

[Prezentarea lucrării 16](#_Toc197604426)

[Capitolui 1. Introducere 18](#_Toc197604427)

[1.1 Introducere în domeniu și aria de utilizare 18](#_Toc197604428)

[1.2 Stadiul actual 20](#_Toc197604429)

[1.3 Obiectivele lucrării 22](#_Toc197604430)

[Capitolul 2. Noțiuni teoretice 23](#_Toc197604431)

[2.1 Analiza semnalului acustic 23](#_Toc197604432)

[2.1.1 Semnalul acustic emis prin dactilografiere 23](#_Toc197604433)

[2.1.2 Arhitectura unei tastaturi 23](#_Toc197604434)

[2.1.3 Analiza în domeniul frecvență 24](#_Toc197604435)

[2.1.4 Coeficienții Mel-Cepstrali 27](#_Toc197604436)

[2.1.5 Transformata Wavelet 29](#_Toc197604437)

[2.2 Algoritmi de învățare profundă 31](#_Toc197604438)

[2.2.1 Rețele neuronale 31](#_Toc197604439)

[2.2.2 Rețele neuronale convoluționale 37](#_Toc197604440)

[2.2.3 CoAtNet 39](#_Toc197604441)

[2.2.4 Procesarea limbajului natural 40](#_Toc197604442)

[2.2.5 Metrici de evaluare a modelului 41](#_Toc197604443)

[2.4 Tehnologii software utilizate 43](#_Toc197604444)

[2.4.1 Python 43](#_Toc197604445)

[2.4.2 Pytorch 44](#_Toc197604446)

[2.4.3 Librosa 44](#_Toc197604447)

[Capitolul 3. Implementarea practică 46](#_Toc197604448)

[3.1 Structura proiectului 46](#_Toc197604449)

[3.2 Descrierea setului de date 47](#_Toc197604450)

[3.3 Analiza semnalelor audio 48](#_Toc197604451)

[3.4 Prelucrarea setului de date 51](#_Toc197604452)

[Bibliografie 52](#_Toc197604453)

# Listă de figuri

[Figura 2. 1 Algoritmul de extragere al coeficienților Mel-cepstrali 28](#_Toc197366077)

[Figura 2. 2 Structura unui neuron artificial 32](#_Toc197366078)

[Figura 2. 3 Arhitectura unei rețele neuronale 34](#_Toc197366079)

[Figura 2. 4 Algoritmul gradientului descendent 36](#_Toc197366080)

[Figura 2. 5 Arhitectura de bază a unei rețele neuronale convoluționale 38](#_Toc197366081)

[Figura 2. 6 Matricea confuziilor 41](#_Toc197366082)

[Figura 3. 1 Etapele necesare realizării proiectului 47](#_Toc197984319)

[Figura 3. 2 Structura setului de date 49](#_Toc197984320)

[Figura 3. 3 Forma de undă pentru tasta „q” 51](#_Toc197984321)

[Figura 3. 4 Forma de undă pentru tasta „m” 51](#_Toc197984322)

[Figura 3. 5 Analiza în domeniul frecvență pentru tasta „q” 52](#_Toc197984323)

[Figura 3. 6 Analiza în domeniul frecvență pentru tasta „m” 52](#_Toc197984324)

[Figura 3. 7 Melspectrograma pentru tasta „q” 53](#_Toc197984325)

[Figura 3. 8 Melspectrograma pentru tasta „m” 53](#_Toc197984326)

[Figura 3. 9 Reprezentarea MFCC a tastei „q” 54](#_Toc197984327)

[Figura 3. 10 Reprezentarea MFCC a tastei „m” 54](#_Toc197984328)

[Figura 3. 11 Reprezentarea Wavelet pentru tasta „q” 55](#_Toc197984329)

[Figura 3. 12 Reprezentarea Wavelet pentru tasta „m” 55](#_Toc197984330)

[Figura 3. 13 Structura CNN 57](#_Toc197984331)

[Figura 3. 14 Structura utilizată pentru modelul CoAtNet 58](#_Toc197984332)

# Listă de tabele

[Tabel 1 Configurația arhitecturală a modelului CoAtNet-0 58](#_Toc197984335)

[Tabel 2 Parametrii de antrenare 61](#_Toc197984336)

# Listă de abrevieri

AI Artificial Intelligence

ML Machine Learning

MFCC Mel-Frequency Cepstral Coefficients

HMM Hidden Markov Model

STFT Short-Time Fourier Transform

SVM Support Vector Machines

CNN Convolutional Neural Network

PBS Polybutylene Terephthalate

ABS Acrylonitrile Butadiene Styrene

PCB Printed Circuit Board

DFT Discrete Fourier Transform

FFT Fast Fourier Transform

STFT Short-Time Fourier Transform

DCT Discrete Cosine Transform

CWT Continuous Wavelet Transform

ANN Artificial Neural Network

ReLU Rectified Linear Unit

SGD Stochastic Gradient Descent

GPU Graphics Processing Unit

NLP Natural Language Processing

PIL Python Imaging Library

# Prezentarea lucrării

Lucrarea este structurată în 5 capitole principale, fiecare prezentând aspecte esențiale pentru realizarea proiectului.

Primul capitol oferă o perspectivă generală a temei, mai exact: introducerea în domeniul studiat și aria de utilizare, prezentarea stadiului actual și obiectivele propuse.

Capitolul 2 aduce în atenție aspectele teoretice necesare înțelegerii metodologiilor adoptate în cadrul proiectului. Acest capitol detaliază algoritmii de prelucrare a semnalului acustic și arhitecturile implementate în contextul sistemului.

Capitolul 3 ilustrează implementarea practică, bazată pe principiile prezentate în capitolul 2. Implementarea poate fi împărțită în mai multe etape: preprocesarea setului de date, implementarea arhitecturilor, atât cât pentru etapa de antrenare, cât și pentru etapa de testare și generarea predicțiilor.

Capitolul 4 furnizează informații despre rezultatele și performanțele obținute de metodele propuse, precum și comparații pe baza acestora. De asemenea, capitolul prezintă dificultățile întâmpinate și metodele pentru soluționarea acestora.

Ultimul capitol expune concluziile obținute în urma realizării proiectului. În plus, sunt propuse direcții de cercetare viitoare pentru a extinde rezultatele obținute.

# Capitolui 1. Introducere

## Introducere în domeniu și aria de utilizare

Dactilografierea poate fi definită ca procesul de scriere a unui text prin utilizarea unei tastaturi. Fiecare tastă produce un sunet unic, iar prin analiza acestor semnale acustice este posibilă determinarea tastelor care au fost apăsate. Sunetele distincte se datorează faptului că tastele sunt poziționate diferit pe placa tastaturii. Studiile indică și existența unor posibili factori secundari, precum influența tastelor alăturate sau diferențele microscopice prezente în construcția tastelor, dar acestea sunt considerate neglijabile. [1]

Aceste semnale acustice pot fi utilizate în cadrul atacurilor de tip canal lateral, mai exact folosind tehnici de procesare a semnalelor și de recunoaștere a tiparelor, astfel compromițând confidențialitatea datelor. Un atac de tip canal lateral este un atac cibernetic care se bazează pe exploatarea implementării fizice pentru obținerea informațiilor sensibile. Aceste informații pot fi divulgate prin diverse metode, cum ar fi: consumul de energie, timpul de execuție, emisiile electromagnetice sau emisiile acustice. [2]

Procesarea semnalelor acustice se referă la analiza acestora pentru a extrage caracteristicile relevante. Analiza semnalelor presupune o serie de etape, precum captarea sunetului cu ajutorul unui microfon, preprocesarea, ultima etapă fiind extragerea caracteristicilor. Datele devin mai ușor de gestionat prin aplicarea procesului de extragere a trăsăturilor, doarece acesta elimină toate caracteristicile ineficiente, fără a pierde informațiile relevante. [3] Ulterior, sunt utilizate diverse metode de recunoaștere a tiparelor pentru identificarea tastelor. Abordările utilizează analiza temporală [4] sau spectrală, care ulterior implică fie procesarea mai detaliată a semnalelor și analiză statistică [5], [6], fie algoritmi de învățare automată, respectiv învățare profundă. [1], [2], [6], [7], [8], [9], [10] Această lucrare se concentrează pe aplicarea tehnicilor de învățare profundă.

Sociologul Sherry Turkle, care își axează cercetarea asupra dinamicii relației dintre om și tehnologie, afirmă că „inteligența artificială este adesea descrisă ca arta și știința de a face mașinile să realizeze lucruri care ar fi considerate inteligente dacă ar fi făcute de oameni.”[11] Deși definiția clasică caracterizează inteligența artificială ca pe o emulație a inteligenței umane, aceasta poate depăși această limitare. Dacă ne focusăm asupra proceselor prin care AI ajunge la obținerea rezultatelor, mai exact asupra mecanismelor de funcționare și a structurilor utilizate, putem observa o perspectivă diferită asupra naturii inteligenței artificiale.

Inteligența artificială reprezintă una dintre cele mai revoluționare tehnologii actuale, având o influență majoră asupra mai multor domenii cum ar fi: finanțele, sănătatea, transporturile, justiția etc. În domeniul financiar, aceasta permite luarea automată a deciziilor privind aprobarea împrumuturilor și realizarea de investiții personalizate prin intermediul robo-consilierilor. În sistemul sanitar, inteligența artificială reușește să îmbunătățească acuratețea diagnosticelor, în special în imagistica medicală și în gestionarea bolilor cronice. Sistemul de justiție utilizează tehnologii bazate pe AI precum recunoaștere facială, un concept tot mai răspândit asemena analizei amprentelor. În transporturi, AI este integrată în vehicule autonome prin intermediul senzorilor și a algoritmilor de învățare profundă care permit vehiculelor navigarea independentă. [12]

Însă pentru a înțelege amploarea transformărilor pe care inteligența artificială le generează în cadrul societății, este important să ne concentrăm asupra evoluției acestui domeniu. Conceptul de inteligență artificială a apărut în timpul celui de-al Doilea Război Mondial, ca urmare a necesității de a dezvolta mașini și tehnologii care să poată soluționa probleme complexe. Învățarea automată este un subdomeniu al inteligenței artificiale care are ca scop dezvoltarea algoritmilor care permit computerelor să învețe și să ia decizii cu ajutorul datelor, fără a fi programate explicit. Câteva subiecte abordate prin prisma conceptului ML sunt: recunoașterea imaginilor și a tiparelor, procesarea limbajului natural, analiza sentimentelor, analiza comportamentului etc. [13] Progresele hardware dar și software, au condus la dezvoltarea conceptului de învățare profundă, un subdomeniu specializat al învățării automate, lucru care indică interdependența dintre aceste tehnologii. Modul de funcționare al acestor algoritmi este bazat pe conceptele fundamentale ale creierului uman, precum neuroni, interconectarea acestora, modul de procesare ierarhic al informațiilor etc. Acești algoritmi pot învăța reprezentări complexe ale datelor și pot recunoaște tipare subtile, ceea ce îi face potriviți pentru sarcina proiectului.

Tema studiată în cadrul lucrării este „Identificarea și clasarea semnalului acustic emis prin dactilografiere”. Motivația dezvoltării în direcția acestei teme este importanța evidențierii vulnerabilităților sistemelor datorate emisiilor generate de implementarea fizică, în special emisia semnalelor acustice.

În 2004, R. Agrawal a demonstrat, pentru prima oară, potențialul exploatării emisiilor acustice ale tastaturii pentru extragerea datelor rezultate in urma tastării [1]. De atunci, numeroase lucrări au fost realizate pentru a îmbunătăți performanțele studiilor anterioare prin diferite metode și în diverse scenarii.

## Stadiul actual

Există mai multe maniere de abordare a temei, fiecare oferind soluții diferite pentru tratarea subiectului.

În publicația [1], menționată anterior, este propusă o metodă care folosește Transformata Fourier aplicată segmentelor audio, pe baza căreia este antrenată o rețea neuronală pentru a recunoaște apăsările tastelor. Sunt prezentate două scenarii principale: distingerea dintre două taste, unde s-a obținut un grad ridicat de acuratețe și distingerea dintre 30 de taste, unde acuratețea maximă atinsă a fost de 79%. De asemenea, studiul explorează și situația în care rețeaua este antrenată cu ajutorul unei tastaturi și testată cu ajutorul altei tastaturi, caz în care tasta corectă s-a aflat în primele 4 predicții în 52% dintre teste. Ultimul experiment a constat în studierea impactului stilurilor diferite de dactilografiere. S-a observat că, atunci când rețeaua este antrenată folosind un set de date care include apăsări de taste efectuate în stiluri diferite, tastele sunt recunoscute cu o acuratețe similiră cazurilor inițiale.

Lucrarea [4] continuă cercetările și implementează o metodă alternativă bazată pe învățare nesupervizată, care nu necesită date de antrenament etichetate. Atacul propus utilizează modele statistice de limbaj, coeficienți Mel-cepstrali (MFCC) și modele Markov (HMM) pentru identificarea și clasarea semnalelor acustice. Studiul demonstrează faptul că folosirea caracteristicilor cepstrale duce la rate de recunoaștere a tastelor superioare în comparație cu metoda Transformata Fourier Rapidă (FFT). Ulterior, a fost introdus un mecanism de învățare supervizată bazat pe feedback, care utilizează rezultatele corecte ca date de antrenament pentru îmbunătățirea performanțelor. Acuratețea maximă obținută în urma procesului de invățăre nesupervizată este 87,195% pentru caractere și 74,57% pentru cuvinte și respectiv, în urma introducerii mecanismului bazat pe feedback 96,34% pentru caractere și 90,46% pentru cuvinte.

Ideile prezentate în [5] oferă o perspectivă nouă asupra temei, implementând o metodă care nu necesită antrenament, capabilă să recunoască cuvinte formate din 7-13 litere. Metoda implică segmentarea semnalului în evenimente de apăsare și eliberare a tastei, măsurarea similarității acustice, utilizând corelația și construirea unei matrice de similaritate. Pe baza acesteia se realizează constrângeri, folosite ulterior pentru clasificarea tastelor. Rata generală de succes este de 73%, fiind direct afectată de lungimea cuvintelor.

Publicația [10] demonstrează un atac care exploatează spectrogramele realizate pe baza sunetelor tastelor și o rețea neuronală simplă formată doar din două straturi. Apăsările tastelor sunt procesate folosind Transformata Fourier pe Termen Scurt (STFT) pentru a crea spectrogramele, valorile frecvențelor fiind introduse în vectori de caracteristici, utilizați pentru antrenarea rețelei. Acuratețea rezultată este 72,3% în experimentele cu 80 de exemple pentru fiecare tastă și a crescut până la 98.8% atunci când experimentele au fost realizate cu un set de date considerabil mai mic.

În lucrarea [2], Mubarek Mohammed explorează mai mulți algoritmi de învățare automată. Studiul formulează două sarcini de clasificare: distingerea tastelor de tip spațiu (space, enter) de litere și identificarea literelor individual. Apăsările tastelor sunt convertite în Melspectrograme, iar pentru extragerea trăsăturilor este folosită rețeaua neuronală pre-antrenată ResNet18. Aceste trăsături sunt ulterior clasate cu ajutorul algoritmilor precum Support Vector Machines (SVM), Random Forest sau XGBoost. Modelul SVM a obținut cele mai bune rezultate, cu o acuratețe de 96% în diferențierea literelor față de tastele de tip spațiu și o acuratețe de 53% în recunoașterea literelor.

Lucrarea [7] introduce o metodă capabilă să recunoască tastele, utilizând algoritmul de învățare profundă CoAtNet, un model care combină straturi convoluționale cu straturi de „self-attention”1. Pentru extragerea caracteristicilor acustice, sunt utilizate Melspectrogramele, deoarece au o structură vizuală compatibilă cu modelul ales în cadrul lucrării. Acuratețea obținută în cazul înregistrărilor realizate cu un telefon este 95%, iar în cazul înregistrărilor efectuate prin intermediul platformei Zoom acuratețea este 93%. Studiul arată fezabilitatea atacului, atât local cu ajutorul unui microfon, cât și remote, folosind platforme de comunicare online.

Un studiu recent [14] propune o abordare, ce constă în construirea unui set de date propriu ce conține 36 caractere alfanumerice și folosirea coeficienților Mel-cepstrali (MFCC) pentru extragerea trăsăturilor audio. În etapa de antrenare, este implementată o rețea neuronală convoluțională (CNN) compusă din straturi de convoluție, normalizare, grupare (pooling), dezactivare a neuronilor (dropout) și straturi dense. Sistemul obține o acuratețe de 94,6% în recunoașterea caracterelor, demonstrând eficiența metodei propuse.

Toate aceste studii evidențiază atât pericolul, cât și practicalitatea atacurilor bazate pe semnalele acustice emise în timpul tastării. Ele arată vulnerabilitatea sistemelor informatice atunci când sunetele tastelor sunt exploatate pentru a deduce informații precum parole. De-a lungul anilor, a existat o evoluție clară în ceea ce privește performanțele sistemelor dezvoltate.

1 Mecanism utilizat în domeniul învățării automate pentru a surprinde dependențe și relații în cadrul secvențelor de intrare

Cu toate acestea, potențialul dezvoltării rămâne considerabil, având în vedere progresele tehnologice din domeniu, fapt ce lasă deschisă oportunitatea de cercetare în această direcție.

## Obiectivele lucrării

Lucrarea are ca obiectiv principal implementarea unui sistem de identificare și clasare a semnalului acustic emis prin dactilografiere. Pentru atingerea acestui scop au fost parcurși mai mulți pași astfel:

*Studiul semnalului acustic emis prin dactilografiere*, în special caracteristicile definitorii și analiza acestuia în vederea extragerii trăsăturilor audio necesare în etapa următoare a proiectului. Se definesc la nivel teoretic caracteristicile semnalului, iar ulterior se implementează la nivel practic mai multe tehnici specifice de procesare audio în vederea extragerii acestor caracteristici.

*Înțelegerea funcționării algoritmilor de învățare profundă potriviți utilizării în cadrul proiectului și implementarea practică a acestora*. Se prezintă modul de funcționare și componentele unei rețele neuronale, utilizate frecvent în procesarea datelor audio. În cadrul proiectului sunt propuse următoarele arhitecturi: rețea neuronală convoluțională și o versiune modificată a algoritmului CoAtNet. Etapele de antrenare și testare vor fi implementate pe baza unui set de date propriu, iar parametrii utilizați în timpul antrenării vor rămâne aceeași, indiferent de arhitectura utilizată. Rezultatele finale obținute vor fi comparate pentru a observa performanțele fiecărei metode aplicate. Comparația va viza atât tipul de caracteristici utilizate, cât și algoritmul de clasificare pentru a determina ce scenariu oferă cea mai mare acuratețe în recunoașterea tastelor.

*Generarea predicțiilor și corectarea acestora prin aplicarea tehnicilor de Procesare a Limbajului Natural.* Se detaliază cum, pe baza modelelor antrenate, sunt generate predicții prin clasificarea semnalelor audio și atribuirea unei taste. Însă, în cadrul acestui proces, pot aparea erori, astfel ducând la apariția unor predicții false, iar pentru a îmbunătăți acest lucru, sunt utilizate tehnici de Procesare a Limbajului Natural, care oferă posibile alternative pentru înlocuirea cuvintelor greșite.

# Capitolul 2. Noțiuni teoretice

## 2.1 Analiza semnalului acustic

### 2.1.1 Semnalul acustic emis prin dactilografiere

Unda acustică este o oscilație mecanică a presiunii care se propagă printr-un mediu (solid, lichiz, gaz sau plasmă) sub forma unui model periodic, transmițând energie de la un punct la altul. Unda acustică transmite sunetul prin stimularea organelor auditive care produc astfel senzația de auz, motiv pentru care este denumită și semnal acustic. Aceste semnale sunt definite în general de 3 caracteristici: lungimea de undă, frecvența și amplitudinea. [15]

Semnalele pot fi clasificate în funcție de mai multe criterii. În ceea ce privește continuitatea, ele se împart în semnale continue și discrete. Din perspectiva comportamentului în timp, există semnale periodice și neperiodice, iar în funcție de predictibilate, acestea pot fi deterministe sau aleatoare. Semnalele neperiodice pot fi împărțite în semnale tranzitorii și aproape periodice. Semnalele acustice emise prin dactilografiere sunt considerate semnale tranzitorii, ce sunt caracterizate de o durată scurtă, de ordinul câtorva milisecunde, fiind în general încadrate în perioade de liniște sau de zgomot de fond. [16]

Semnalele acustice emise prin apăsarea tastelor pot fi caracterizate prin două regiuni distincte: de apăsare a tastei și respectiv de eliberare. Studiile arată că în funcție de forța cu care este apăsată tasta, atât regiunea de apăsare, cât și cea de eliberare prezintă, în reprezentarea în domeniul timp, între 1-3 vârfuri proeminente. [6] Aceste semnale sunt caracterizate de o amplitudine ridicată în momentul apăsării tastei, care ulterior scade rapid. În domeniul frecvenței, sunetele tastelor sunt concentrate în intervalul jos-mediu al frecvențelor, acest lucru fiind influențat de factori precum stilul dactilografierii forța apăsării sau poziția în care este apăsată tasta. [7]

### 2.1.2 Arhitectura unei tastaturi

Componentele unei tastaturi lucrează împreună pentru a converti apăsări ale tastelor în semnale electrice pentru a fi interpretate de un computer. Fiecare element are un rol important și poate influența direct performanța utilizării tastaturii. În contextul semnalelor acustice, componentele care pot influența caracteristicile sunetului sunt capacele tastelor și comutatoarele.

Capacele tastelor reprezintă componenta situată deasupra comutatoarelor și care intră în contact direct cu degetele utilizatorului. Ele pot influența sunetul emis la apăsare în funcție de materialul din care sunt confecționate. Cele mai comune materiale sunt PBT sau ABS, dar există totuși posibilitatea utilizării materialelor precum rășină, lemn sau aluminiu. De asemenea, capacele pot avea profiluri diferite și se disting prin variații de înălțime, formă și geometria rândurilor de taste.[17]

Comutatoarele tastaturilor sunt unele dintre cele mai importante componente ale tastaturii, deoarece ele determină sunetul tastelor. Acestea prezintă o serie de caracteristici: forța necesară pentru apăsare, senzația apăsării, distanța de cursă a tastei, tipul de iluminare etc. Există trei tipuri de comutatoare care oferă o experiență diferită a tastării: clicky (feedback tactil insoțit de un sunet puternic), tactil (feedback tactil însoțit de un sunet atenuat) și liniar (apăsare mai silențioasă). Există și tastaturi, adesea întâlnite pe laptop, care folosesc comutatoare din cauciuc ce nu oferă feedback tactil și sunt silențioase. [17]

De asemena, tastele de dimensiuni mai mari, cum ar fi Space sau Enter, dispun de un mecanism diferit de funcționare care include stabilizatoare menite adaptării la suprafața extinsă. Aceste stabilizatoare conduc la generarea unui sunet distinct la apăsare în comparație cu tastele standard.[17]

Aceste elemente contribuie la diversitatea tastaturilor existente pe piață. Din acest motiv, apare necesitatea extinderii experimentelor în cadrul procesului de recunoaștere a tastelor pentru a asigura acuratețea și robustețea sistemelor.

### 2.1.3 Analiza în domeniul frecvență

Majoritatea algoritmilor de învățare profundă utilizează caracteristicile obținute în urma analizei în domeniul frecvență, iar acest principiu se aplică și în cadrul proiectului de față. În domeniul timp, semnalul este reprezentat de o formă de undă ce ilustrează cum variază amplitudinea în timp. Reprezentările acestor semnale acustice în domeniul timp nu evidențiază diferențele dintre taste suficient, ceea ce le face mai puțin adecvate pentru antrenarea unor algoritmi de învațare automată. În schimb, reprezentările în domeniul frecvență ilustrează mai bine semnalele și evidențiază variații între taste, contribuind astfel la performanțe mai bune ale modelelor. [10]

Cea mai utilizată tehnică prin care se realizează trecerea unui semnal din perspectiva temporală în cea spectrală este Transformata Fourier.

Transformata Fourier este o tehnică matematică fundamentală utilizată pentru analiza conținutului spectral al semnalelor. În cazul analizei semnalelor digitale, este folosită Transformata Fourier Discretă (DFT). Cu alte cuvinte, Transformata Fourier Continuă se aplică semnalelor analogice, iar DFT operează pe semnale discrete, adică eșantionate și cuantificate, așa cum sunt reprezentate în sistemele digitale. În domeniul procesării audio, este adesea folosită Transformata Fourier Rapidă (FFT), o metodă eficientă pentru calcularea DFT, care descompune semnalul reprezentat în timp în componente spectrale, ilustrând astfel amplitudinea și faza fiecărei componente. [18]

În primul rând, este important de menționat că semnalul acustic emis în timpul dactilografierii are o natură continuă. Pentru a putea fi procesat digital, acesta trebuie convertit într-un semnal discret, fapt ce implică eșantionarea undei sonore la intervale regulate de timp. Rata de eșantionare, exprimată în Hz, reprezintă numărul de eșantioane într-o secundă și determină rezoluția semnalului digital. O rată de eșantionare mai mare permite captarea mai precisă a detaliilor acustice. [18]

(2. 1)

Ecuația (2.1) translatează o secvență de *N* numere complexe *x[n]* din domeniul timp într-o secvență de *N* numere complexe *X[k]* în domeniul frecvență. FFT reprezintă o metodă optimizată care oferă aceleași rezultate ca DFT, dar care necesită mult mai puține operații de calcul. Această creștere a eficienței se datorează descompunerii recursive a Transformatei Fourier Discrete în transformate de dimensiuni mai mici, exploatând proprietățile de periodicitate și simetrie a funcțiilor exponențiale complexe. [18]

Dimensiunea unui cadru se referă la numărul de eșantioane folosite în fiecare operație FFT și influențează semnificativ echilibrul dintre rezoluția în frecvență (superioară în cazul cadrelor mai mari) și rezoluția în timp (mai bună în cazul cadrelor mai mici). [18] Din cauza procesului de fragmentare a semnalului, pot apărea discontinuități la capetele cadrelor, care pot duce la distorsiuni. Pentru atenua acest efect, se aplică ferestre care au rolul de a netezi marginile fiecărui cadru, reducând astfel variațiile bruște. Fereastra Hann este adesea întâlnită în procesarea semnalelor, în special analiza audio, iar prin aplicarea sa pe fiecare cadru, aceasta contribuie la obținerea unor reprezentări spectrale clare și precise. [18], [19]

În practică, algoritmul FFT impune ca dimensiunea ferestrei să fie o putere a numărului 2 pentru a optimiza viteza de procesare. După segmentarea semnalului în cadre, se aplică FFT fiecărui cadru în parte, generând o succesiune de reprezentări spectrale locale, care reflectă evoluția conținutului în frecvență a semnalului de-a lungul timpului. [18]

Semnalul acustic emis prin apăsarea tastelor este un semnal nestaționar, caracterizat prin frecvențe și amplitudini care variază în timp. Spre deosebire de semnalele simple, cum ar fi cele sinusoidale sau rectangulare, semnalele nestaționare nu urmează tipare clare, ceea ce le face mai dificil de analizat. Cele mai eficiente metode de analiză spectrală în acest context sunt STFT sau Transformata Wavelet, ambele capabile să ilustreze variațiile spectrale ale semnalului în funcție de timp. [20]

Transformata Fourier pe Termen Scurt reprezintă o versiune modificată a Transformatei Fourier, concepută pentru a analiza conținutul în frecvență al unui semnal în intervale scurte de timp. Aceasta presupune segmentarea semnalului în porțiuni suprapuse și aplicarea FFT fiecărei porțiuni individuale. STFT este potrivită pentru analiza semnalelor nestaționare, deoarece presupune că, pe intervale mici, un semnal nestaționar se comportă ca un semnal staționar. Principala limitare a STFT este utilizarea ferestrelor de lungime fixă ceea ce înseamnă rezoluție constantă atât în timp, cât și în frecvență. [20]

Rezultatul aplicării STFT este o reprezentare timp-frecvență cunoscută sub numele de spectrogramă. Spectrograma reprezintă imaginea fundamentală a spectrului de frecvențe, evidențiind modul în care componentele de frecvență fluctuează în timp. Pe axa absiceselor este reprezentat timpul, pe axa ordonatelor este reprezentată frecvența, iar intensitatea culorilor indică amplitudinea fiecărei componentă spectrală la momente specifice. De obicei, axa frecvenței este reprezentată pe o scară logaritmică, iar valorile amplitudinii sunt exprimate în decibieli. [21]

Scara Mel reprezintă o transformare logaritmică a frecvențelor semnalelor, concepută pentru a reflecta modul în care urechea umană percepe sunetele. Principiul fundamental este ca distanțele egale pe scara Mel să corespundă percepției umane a distanțelor egale între sunete. Studiile arată că oamenii sunt mai buni la detectarea frecvențelor joase în comparație cu cele înalte. De exemplu, este mai ușor să detectăm diferența dintre 100 și 200 Hz, decât dintre 10100 și 10200 Hz, cu toate că distanța numerică este acceași. [22]

(2. 2)

Formula (2.2) descrie conversia frecvențelor din scara Hz în scara Mel. În această ecuație, logaritmul utilizat se referă la cel natural (logaritm în baza e). Dacă s-ar fi folosit logaritm în baza 10, atunci coeficientul din formulă ar fi avut o valoarea diferită. [22]

Melspectrogramele reprezintă o versiune adaptată a spectrogramelor clasice, în care frecvențele sunt ilustrate pe scara Mel. Această transformare comprimă informația spectrală, oferind o reprezentare mai relevantă din punct de vedere perceptiv. Datorită acestei caracteristici, melspectrogramele sunt adesea utilizate în sistemele de DL, deoarece oferă informații compacte, reducând dimensiunea datelor de intrare și generează reprezentări semnificative din care algoritmii pot învăța ușor. [22]

Aceste reprezentări, Melspectrogramele, pot oferi o bază solidă pentru extragerea caracteristicilor utile în cadrul algoritmilor de învățare profundă. Dar, în plus, pornind de la operația fundamentală care permite trecerea din domeniul timp în domeniul frecvență, transformata Fourier, pot fi aplicate prelucrări suplimentare pentru a extrage și alte trăsături specifice ale semnalului acustic.

### 2.1.4 Coeficienții Mel-Cepstrali

Deși coeficienții Mel-cepstrali sunt concepuți pentru a prioritiza frecvențele relevante în vorbirea umană, aceștia au fost utilizați în numeroase studii pentru extragerea trăsăturilor acustice în contextul recunoașterii tastelor. [4], [9], [14]

Coeficienții Mel-cepstrali sunt derivați din Melspectrograme, dar implică câteva etape de procesare suplimentare, deoarece aceștia reprezintă o reprezentare mai compactă a trăsăturilor spectrale. MFCC sunt calculați urmând 5 pași succesivi: pre-accentuarea semnalului pentru a amplifica frecvențele înalte, segmentarea și aplicarea unei ferestre, aplicarea Transformatei Fourier Rapidă, filtrarea cu ajutorul scării Mel și, în final, aplicarea Transformatei Cosinus Discret (DCT). [3]

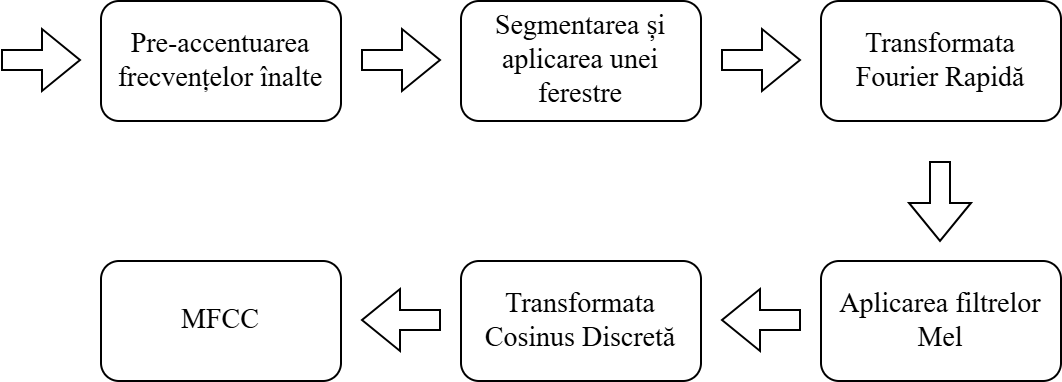


Figura 2. 1 Algoritmul de extragere al coeficienților Mel-cepstrali

Pre-accentuarea este un pas important în compensarea pierderilor din componentele de frecvență înaltă ale semnalului care pot apărea în timpul generării semnalului. Această operație constă în aplicarea unui filtru trece-sus, care are scopul de a amplifica frecvențele înalte, adică acele componente care se schimbă rapid în timp. [23]

Ulterior, semnalul este împărțit în mai multe cadre pentru a aplica FFT. Dimensiunea cadrelor poate varia între 25 ms și până a 500 ms. De asemenea, un parametru important este cel care indică numărul de eșantioane dintre începuturile cadrelor consecutive. Dacă acesta egal cu dimensiunea unui cadru, atunci nu va exista suprapunere între cadre. Această suprapunere a cadrelor poate ajuta la reducerea pierderilor de informație de la marginea cadrelor, astfel contribuind la o reprezentare mai continua a caracteristicilor. Urmează aplicarea unei ferestre Hann asupra fiecărui cadru și ulterior aplicarea Transformatei Fourier Rapide. [3], [23]

Tranziția către domeniul spectral a fost prima etapă importantă în obținerea coeficienților Mel-cepstrali. Următoarea etapă constă în aplicarea unui set de filtre trece-bandă, cunoscute sub denumirea de filtre Mel. Aceste filtre au fost initial dezvoltate pentru analiza vorbirii și, asemenea modului în care urechea umană percepe sunetele, acestea au ca scop extragerea unei reprezentări neliniare a semnalului. Filtrele implementate au formă triunghiulară și sunt, de regulă în număr de 40. Scopul lor este de a obține valorile corespunzătoare frecvențelor centrale, întrucât fiecare filtru atinge valoarea 1 în dreptul frecvenței centrale și descrește linear către 0, aceasta fiind frecvența centrală dintre două filtre adiacente. Procesul de prelucrare implică înmulțirea spectrului semnalului cu fiecare dintre filtrele Mel, continuată de transpunerea pe o scară logaritmică și deducerea coeficienților. Funcția de transfer a unui filtru poate fi calculate astfel: [24] [3]

(2. 3)

unde, este frecvența centrală a filtrului și .

Ultimul pas înainte de obținerea coeficienților Mel-cepstrali este aplicarea Transformatei Cosinus Discretă. DCT exprimă o secvență finită de date ca o sumă de funcții cosinus care oscilează la diferite frecvențe. Mai exact, readuce valorile în domeniul frecvență În procesul obținerii MFCC, DCT se aplică bancului de filtre Mel pentru a selecta cei mai importanți coeficienți. DCT se calculează astfel: [25]

(2. 4)

Semnalul este transformat într-o reprezentare cepstrală, rezultatul fiind un set de coeficienți care alcătuiesc o imagine spectrală compactă a semnalului.În literatura de specialitate, cele mai multe lucrări propun metoda standard de calcul a MFCC, descrisă anterior, folosind 13 sau mai puțini coeficienți, considerați suficienți pentru a capta caracateristicile semnalului. [3]

### 2.1.5 Transformata Wavelet

Așa cum a fost menționat anterior, semnalele acustice emise prin dactilografiere sunt semnale nestaționare, fiind caracterizate prin variații rapide ale amplitudinii și componente de frecvență variabile în timp. Din acest motiv, o metodă eficientă de analiză a acestora este reprezentată de Transformata Wavelet.

Un dezavantaj major al transformatei Fourier este că oferă o reprezentare spectrală globală, evidențiind frecvențele care persistă pe întreaga durată a semnalului, fără a furniza informații despre momentul apariției acestora. Transformata Wavelet permite o analiză simultană în timp și în frecvență, oferind o reprezentare mai detaliată. Deși STFT reprezintă o alternativă viabilă pentru analiza semnalelor nestaționare, Transformata Wavelet este adesea preferată, deoarece STFT implică segmentarea semnalului iar apoi aplicarea FFT asupra fiecărui segment, limitând astfel flexibilitatea rezoluției. [26]

Transformata Wavelet reprezintă o tehnică matematică folosită pentru a descompune un semnal în componente obținute prin scalarea și translatarea unei funcții denumită wavelet. Transformata Wavelet Continuă (CWT) a unui semnal se definește ca: [27]

(2. 5)

În ecuația (2.5), *ψ(t)* reprezintă funcția wavelet, aleasă în funcție de caracteristicile semnalului analizat. Cele mai utilizate exemple sunt: Morlet, folosită pentru semnale oscilatorii, Mexican Hat, ideală pentru evenimente tranzitorii, și Haar, potrivită pentru detecția marginilor. Parametrul *a* reprezintă factorul de scalare, pentru care valorile mari corespund frecvențelor joase, iar valorile mici evidențiază frecvențele înalte. Parametrul *b* este factorul de translatare, care caracterizează poziționarea waveletului în timp: scăderea valorii îl deplasează spre stânga, iar creșterea îl deplasează spre dreapta. Acest parametru este important, deoarece funcțiile wavelet sunt localizate în timp și diferite de valoarea 0 în intervale scurte. Ideea de bază a funcționării constă în determinarea gradului de corespondență dintre semnalul analizat și o funcție wavelet pentru anumite valorile pentru scară și locație. Acest lucru se realizează prin convoluția semnalul cu un set de funcții wavelet, fiecare având parametri specifici. Rezultatul operației de convoluție ne oferă un coeficient, care indică cât de bine se potrivește waveletul cu semnalul la o anumită scară într-un anumit moment de timp. [26], [27]

Rezultatul obținut în urma aplicării Transformatei Wavelet este o reprezentare timp-frecvență a semnalului, ce ne arată cum evoluează componentele spectrale în timp. [27]

Funcția wavelet Morlet este o componentă fundamentală a analizei timp-frecvență, inițial concepută pentru procesarea semnalelor seismice. Este definită ca o undă sinusoidală modulată printr-o anvelopă Gaussiană2. În cadrul CWT, acest tip de wavelet este scalat și translatat pentru a evidenția caracteristici la diferite rezoluții. O trăsătură definitorie a acestui wavelet este prezența unui parametru care

2 O curbă în formă de clopot, similară unei distribuții normale, utilizată pentru a modula o altă funcție

permite ajustarea dimensiunii ferestrei Gaussiene în raport cu perioada semnalului sinusoidal. [28]

## 2.2 Algoritmi de învățare profundă

Înainte de a aprofunda conceptul de învățare profundă, este importantă înțelegerea principiile fundamentale ale învățării automate. Algoritmii de ML analizează seturi de date complexe și identifică tipare pe care le folosesc pentru a genera predicții sau recomandări. Performanța acestor algoritmi se îmbunătățește pe măsură ce sunt expuși la volume mai mari de date. Creșterea exponențială a volumului și complexității datelor, dincolo de capacitățile de procesare umane, evidențiază importanța acestei tehnologii în diverse aplicații. Metodele de învățare automată erau limitate în ceea ce privește capacitatea lor de a lucra cu date brute, mai exact pentru a dezvolta un sistem de recunoaștere a tiparelor era necesară intervenția umană pentru a extrage caracteristici și a transforma datele într-un format potrivit. Pentru a miniza această dependență, a fost dezvoltat domeniul învățării profunde capabil să învețe caracteristici direct din date brute, precum imagini, text sau audio. Deep learning este bazat pe conceptul de rețele neuronale artificiale (ANN), inspirate din modul în care funcționează creierului uman. Aceste rețele sunt formate din mai multe straturi de neuroni interconectați care procesează informația în etape succesive. Fiecare etapă determină sistemul să recunoască trăsături din ce în ce mai complexe și mai abstracte. De exemplu, un strat inițial poate detecta margini sau forme simple, în timp ce straturi mai profunde pot îmbina aceste forme pentru a identifica obiecte sau concepte. Asemenea ML, sistemele DL învață prin iterații, denumite epoci, ajustându-și parametrii pentru îmbunătățirea continuă a performanțelor. [29], [30]

### 2.2.1 Rețele neuronale

Creierul uman, ca sursă de inspirație, a determinat dezvoltarea sistemelor artificiale de procesare a informației, precum neuronii artificiali. El este alcătuit din aproximativ 86-100 de miliarde de neuroni interconectați, aceștia fiind specializați în recepționarea, procesarea și transmiterea semnalelor electrice. Numărul de conexiuni sinaptice este de aproximativ 600 de ori mai mare, indicând importanța intedependenței dintre neuroni pentru o funcționare corespunzătoare. O rețea neuronală artificială este o structură artificială bazată pe modul de funcționare a creierului uman, fiind alcătuite din multiple elemente de procesare sau neuroni artificiali organizați în straturi. [31]

Una dintre cele mai simple forme a ANN este perceptronul, dezvoltat de Frank Rosenblatt în anii 1950-1960, care s-a inspirat din studiile lui Warren McCulloch și Walter Pitts. Aceștia au reușit în anul 1943, implementarea primului model neuronal artificial care funcționa ca un dispozitiv binar. Acesta stă la baza dezvoltării unor arhitecturi mult mai complexe, ce sunt alcătuite din mai multe straturi neuronale. [31], [32]

Un neuron artificial funcționează prin preluarea mai multor intrări binare și produce o singură ieșire. Fiecăre intrare este asociată cu o pondere sinaptică, , care reflectă importanță datelor de intrare respective în decizia finală. Semnalul de intrare este procesat prin înmulțirea cu valorile ponderilor asociate, iar ulterior are loc o însumare globală a rezultatelor obținute. Sumei rezultate i se adaugă o constantă (bias) care contribuie la decizia de activare a neuronului. Ulterior, este aplicată o funcție de activare și este transmis mai departe un răspuns. [31], [32]

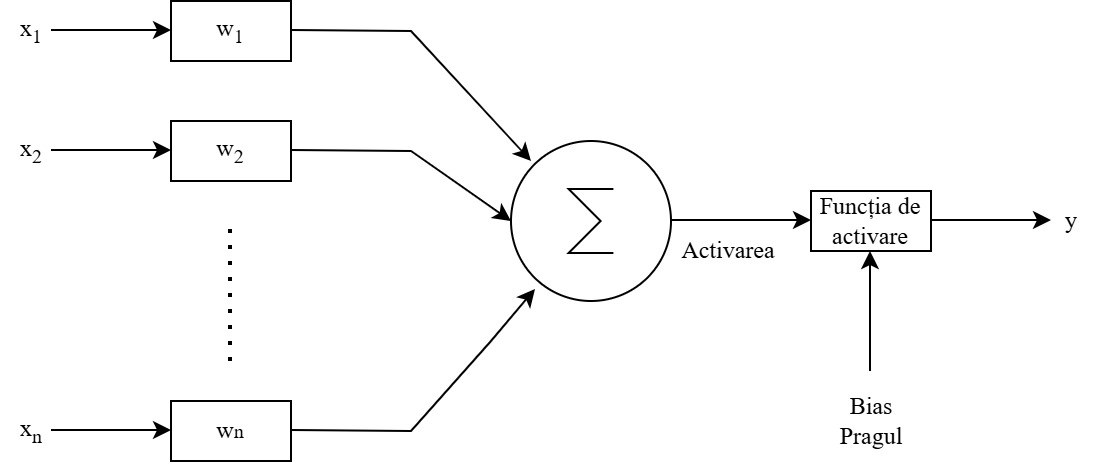


Figura 2. 2 Structura unui neuron artificial

Relația care caracterizează structura neuronului din Figura 2.2 este:

(2. 6)

Funcția *f*  reprezintă funcția de activare, cea care decide dacă un neuron ar trebui „activat” sau nu, în funcție de datele de intrare. Mai exact, se referă la o operație matematică aplicată sumei rezultate care decide dacă datele sunt suficient de relevante pentru a influența rezultatul final. Importanța lor constă în faptul că adaugă neliniaritate modelului, o proprietate esențială în procesul de învățare al rețelei. Fără aceste funcții matematice, modelul ar fi alcătuit doar din operații liniare, ceea ce ar face imposibilă detectarea unor dependențe complexe dintre date. [33]

Există mai multe tipuri de funcții matematice de activare care deservesc scopuri diferite în cadrul rețelelor neuronale. Funcția de activare binară (Binary step function) reprezintă o funcție ce acționează pe baza unui prag impus pentru activarea neuronilor. Totuși, aceasta are o limitare semnificativă, mai exact, capacitatea de a produce doar un rezultat binar, ceea ce o face inadecvată pentru problemele de clasificare multi-clasă. Funcția de activare liniară nu aplică niciun fel de transformare asupra datelor, ceea ce limitează capacitatea de învățare a rețelei și face ca toate straturile să se rezume la operații liniare, anulând avantajele unei rețele profunde. Toate aceste limitări pot fi depășite prin utilizarea funcțiilor neliniare, care permit modelelor să învețe relații complexe și neliniare dintre datele de intrare și cele de ieșire. [33]

Sigmoid reprezintă o funcție de activare care transformă datele de intrare în valori cuprinse în intervalul (0,1), ceea ce o face potrivită pentru modele care trebuie să exprime datele de ieșire sub forma unor probabilități. Pentru valorile de intrare situate în afara intervalului (-3,3), procesul de învățare poate fi încetinit sau chiar blocat. În plus, deoarece intervalul nu este centrat în jurul valorii 0, toate activările au același semn, lucru ce poate provoca instabilitate în antrenarea rețelei. Tangenta hiperbolică este asemănătoare funcției sigmoid, diferența fiind că datele sunt convertite în valori cuprinse între (-1,1), un interval centrat în 0, astfel oferind o învățare mai eficientă. ReLU (Rectified Linear Unit) este o funcție de activare care returnează valoarea de intrare dacă acestea este pozitivă și 0 dacă aceasta este negativă. Prin activarea doar unui set de neuroni, ReLU este eficientă din punct de vedere al calculelor de procesare, fiind una dintre cele mai utilizate funcții în rețele neuronale. Însă, pentru valori negative, există riscul apariției „neuronilor morți”, care nu sunt activați niciodată, un lucru care afectează performanța modelului. Funcția Softmax este o extensie a funcției Sigmoid adaptată pentru clasificarea multi-clasă, convertind rezultatele în valori probabilistice care se însumează la 1. De regulă, Softmax este folosit în stratul final al unei rețele, deoarece permite identificarea clasei prin atribuirea unei probabilități maxime unei ieșiri, astfel oferind o predicție finală. [33]

Alegerea unei funcții de activare într-o reța poate influența semnificativ performanțele acesteia trebuie realizată ținând cont de tipul de predicție dorit și de arhitectura rețelei. În general, funcția ReLU este recomandată pentru utilizarea în cadrul straturilor intermediare, fiind adesea utilizată în rețelele neuronale convoluționale. Pe de altă parte, Tangenta hiperbolică și Sigmoid sunt mai des folosite în cadrul rețelelor neuronale recurente. În ceea ce priveșe stratul de ieșire, alegerea funcției depinde de natura sarcinii: pentru clasificare binară se utilizează funcția Sigmoid, regresia are nevoie de o funcție liniară și clasificarea multi-clasă necesită funcția Softmax. [33]

Rețelele neuronale artificiale sunt organizate, la un nivel de bază, în straturi, care reprezintă o componentă importantă în arhitectura unei rețele. Un strat este format din mai mulți neuroni care procesează datele în paralel. Cele mai utilizate rețele includ trei tipuri de straturi: stratul de intrare, stratul intermediar (ascuns) și cel de ieșire. Stratul de intrare constă în neuroni care preiau datele brute și inițiază procesul de învățare al rețelei. Straturile intermediare realizează majoritatea calculelor, facilitând astfel învățarea. În final, stratul de ieșire generează rezultatele și oferă predicțiile finale ale rețelei. [34]

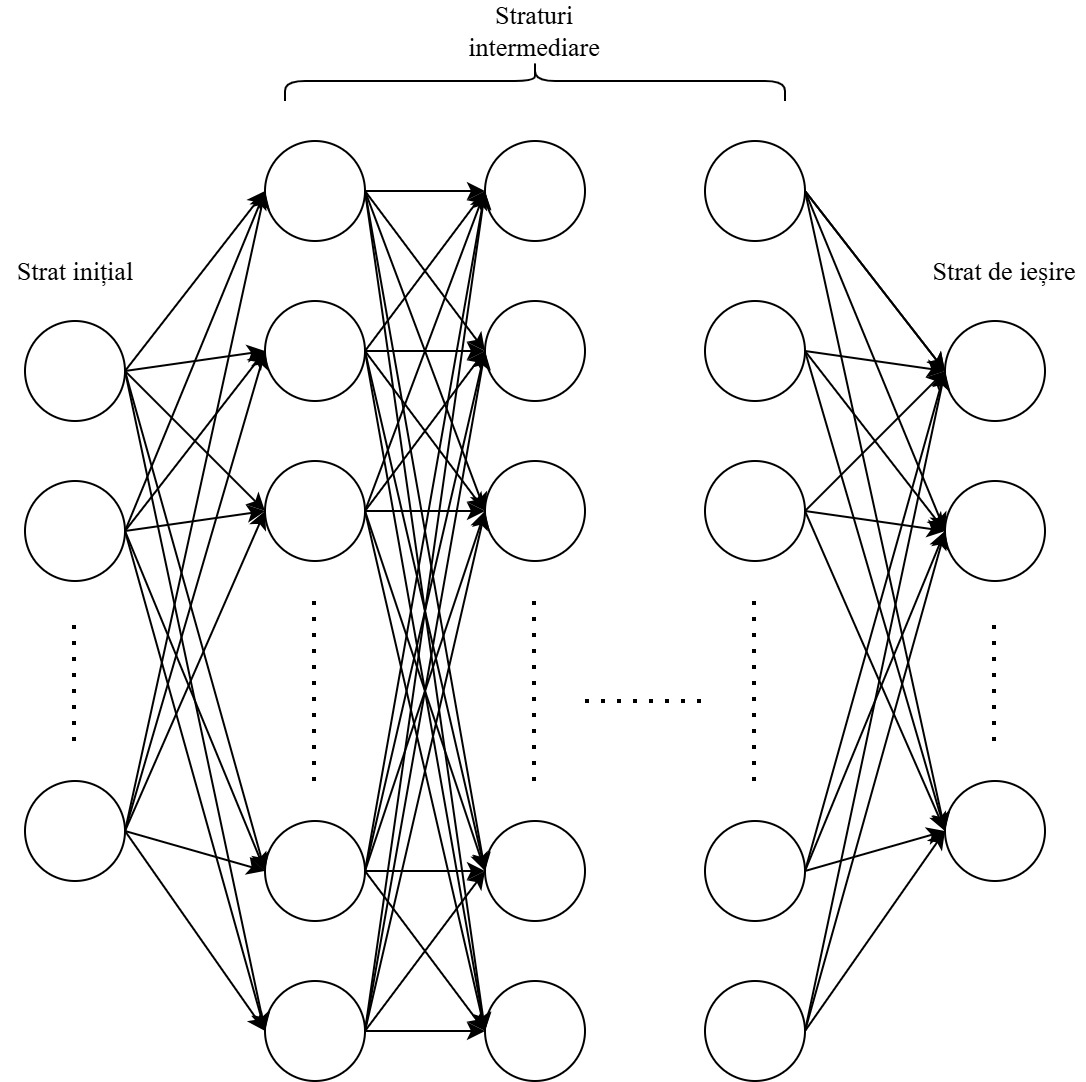


Figura 2. 3 Arhitectura unei rețele neuronale

Învățarea într-o rețea neuronală reprezintă procesul prin care rețeaua își ajustează automat valorile ponderilor pe baza unui set de date de antrenare, fără a fi necesară intervenția manuală asupra fiecărui parametru. Scopul principal este de a obține fenomenul de generalizare, adică abilitatea rețelei de a oferi predicții corecte pentru date care nu au fost întâlnite în timpul antrenării. În cadrul învățării supervizate, algoritmul compară rezultatul generat de rețea și rezultatul dorit, care este furnizat rețelei prin etichetarea corespunzătoare a datelor de intrare. Diferența dintre cele două valori reprezintă eroarea de adaptare care este utilizată ulterior pentru a corecta ponderile. Ajustările sunt realizate, folosind algoritmi de antrenare, care au scopul de a minimiza eroarea, astfel îmbunătățind precizia modelului. [31]

Procesul fundamental care stă la baza învățării supervizate este algoritmul „backpropagation”. Acest algoritm aplică regula lanțului din calculul diferențial pentru a determina modul în care ajustarea parametrilor rețelei afectează eroarea totală, definită cu ajutorul unei funcții de cost. La nivel matematic, obiectivul principal este calcularea gradientului funcției de cost în raport cu fiecare parametru. [35]

Acest proces începe cu propagarea în direcția înainte, etapa care generează predicțiile pe baza datelor de antrenament, acestea trecând succesiv prin straturile care compun strucutura rețelei. Mai exact, propagarea în direcția înainte se referă la o succesiune de calcule matematice, în care ieșirea fiecărui strat devine intrarea pentru următorul strat. La începutul procesului de antrenare, valorile parametrilor, weights și bias, sunt inițializați aleator, în timp ce hiperparametrii precum: numărul de straturi, numărul de neuroni și funcțiile de activare sunt setați anterior și rămân neschimbați pe parcursul procesului. În timpul parcurgerii, un eșantion de intrare este introdus în stratul inițial, iar fiecare neuron transmite aceasta valoare, multiplicată cu valoarea unei ponderi, straturilor intermediare. Acestea calculează suma ponderată a datelor, aplică funcția de activare, iar rezultatul este transmis următorului strat. Acest flux se continuă de la stânga la dreapta, până la stratul de ieșire, unde predicția finală este realizată. Funcția de cost transformă eroarea obținută într-o valoare numerică care reflectă tipul și amploarea erorii. Există mai multe tipuri de funcții de cost, fiecare potrivită pentru diverse sarcini. Funcția Eroare Medie Pătratică este frecvent utilizată pentru regresii, în timp ce unei sarcini de clasificare îi este potrivită funcția de pierdere „cross-entropy”. [36]

Propagarea pe direcția înapoi se desfășoară în direcția opusă propagării înainte, începând cu ultimul strat al rețelei. În această etapă, funcția de cost este derivată pentru a observa cum fiecare parametru contribuie la eroarea totală. Scopul final al clasificării este ca rețeaua să fie ajustată astfel încât neuronul care corespunde clasei corecte să genereze o valoare apropiata de 1, iar celelalți neuroni valori apropiate de 0. Procesul include calcularea derivatelor parțiale atât în raport cu ponderile, cât și în raport cu valorile de activare ale neuronilor. Rețeaua calculează un set de gradienți, care reprezintă modul în care variații ale parametrilor afectează eroarea. [36]

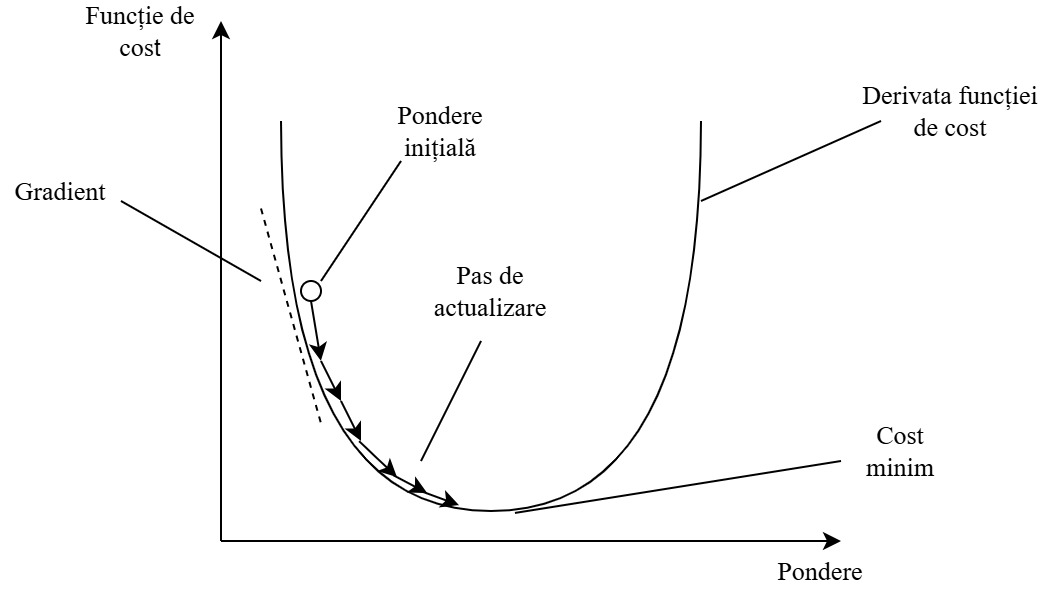


Figura 2. 4 Algoritmul gradientului descendent

Odată ce gradientul funcției de cost a fost obținut, modelul este optimizat folosind metoda gradientului descendent. Acest algoritm funcționează prin ajustarea parametrilor în direcția opusă gradientului, ceea ce duce la o reducere în în cadrul pierderilor. Gradientul calculat conține derivatele parțiale, permițând determinarea direcției corecte în care fiecare parametru trebuie modificat. Fiecare actualizare reprezintă un pas de învățare în cadrul procesului de antrenare. Scopul acestui proces iterativ este de a atinge o valoarea minimă a funcției de cost. [37]

Rata de învățare este un hiperparametru care determină mărimea fiecărui pas de actulizare în timpul algoritmului gradientului descendent. Deoarece rețelele neuronale se bazează pe funcții de activare neliniare, gradientul rezultat poate avea forme complexe, în care un pas inițial în direcția corespunzătoare poate deveni ineficient dacă este prea mare. O rată de învățare mică permite actualizări precise și reduce riscul de a depăși soluția optimă, dar necesită un număr mai mare de iterații, astfel procesul de învățare fiind unul mai lent. O rată de învățare mai mare accelerează învățarea, dar există un risc ridicat de depășire a minimului funcției de pierdere, astfel împiedicând convergența modelului. [36]

Un aspect important al algoritmului gradientului descendent este frecvența cu care sunt actualizați parametrii modelului. Folosind metoda Batch Gradient Descent, gradienții sunt calculați pentru întreg setul de date de antrenament. Acest lucru poate fi ineficient în cazul seturilor de date foarte mari deoarece implică un volum ridicat de calcule. Medoda Stochastic Gradient Descent (SGD) presupune realizarea actualizării pe baza unui singur eșantion. Deși mai rapidă, această metodă poate introduce mai multe fluctuații în cadrul funcții de pierdere. Mini-Batch Gradient este o alternativă care împarte setul de date în grupuri de dimensiune fixă. Această metodă reduce atât cerințele de memorie și timpul de procesre, cât și instabilitatea prezentă în SGD. [37]

În timpul procesului de antrenare, setul de date este împărțit în 3 subseturi: antrenare, validare și testare. Setul de antrenare cuprinde cel mai mare procent din totatul datelor și este utilizat pentru ajustarea ponderilor, astfel încât rețeaua să învețe tiparele din date. Setul de validare are rolul de a observa capacitatea de generalizare a modelului și contribuie la selecția arhitecturii optime sau hiperparametrilor necesari pentru obținerea celor mai bune performanțe. Setul de test este folosit după finalizarea antrenării, pentru evaluarea performanțelor reale ale rețelei. În cadrul modelului poate apărea fenomenul de „underfitting”, atunci când rețeaua își pierde capacitatea de a învăța corespunzător. De asemenea, există și fenomenul de „overfitting”, atunci când modelul învăță bine din datele antrenament, dar nu are performanțe bune pe datele de validare. Aceste probleme pot fi controlate prin ajustarea corespunzătoare a hiperparametrilor sau utilizarea tehnicilor de regularizare. [31], [38]

### 2.2.2 Rețele neuronale convoluționale

Rețelele neuronale convoluționale (CNN) funcționează similar rețelelor neuronale artificiale, fiind alcătuite din neuroni care învață și își optimizează performanțele prin expunerea la date. Diferența majoră este faptul că rețelele convuluționale sunt specializate pentru procesarea imaginilor, deoarece arhitectura este mai eficientă în recunoașterea tiparelor din informații vizuale. Arhitectura acestora încorporează filtre și operații de grupare, care le permit rețelelor să surprindă caracteristici spațiale relevante și să reducă numărul parametrilor. Din acest motiv, CNN sunt eficiente în sarcini precum clasificarea imaginilor, detecția obiectelor sau recunoașterea facială, dar pot fi aplicate și în prelucrarea altor tipuri de date, cum ar fi semnale audio sau vocale. [39]

CNN sunt structurate pe baza a trei tipuri de straturi principale: de convoluție, de grupare și complet conectate. Stratul convoluțional este primul strat și este responsabil de extragerea trăsăturilor simple. Pe măsură ce datele sunt transmise prin straturi convoluționale adiționale și straturi de grupare, rețeaua reușește identificarea unor caracteristici mai complexe. Stratul final este cel complet conectat, care utilizează trăsăturile extrase pentru a realiza clasificarea finală a datelor procesate. [40]

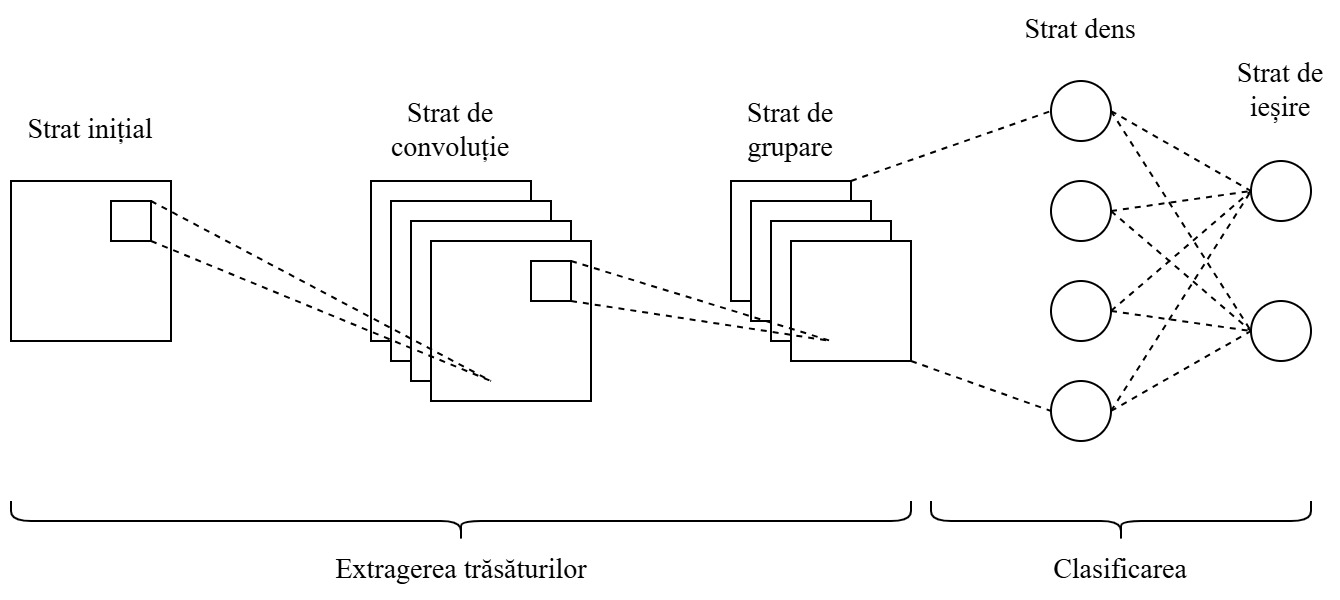


Figura 2. 5 Arhitectura de bază a unei rețele neuronale convoluționale

Stratul de convoluție reprezintă componenta esențială în cadrul CNN. Acesta procesează datele de intrare cu ajutorul filtrelor/nucleelor care se deplasează peste imagine pentru a detecta trăsături. Fiecare filtru, reprezentat de o matrice alcătuită din ponderi, realizează produsul scalar cu valorile corespunzătoare datelor de intrare pentru a genera o „hartă” a caracteristicilor. Acest proces este influențat de hiperparametri: numărul de filtre (care determină profunzimea rezultatului), pasul cu care se deplasează filtrul și padding (pentru a gestiona marginile în cazul unei imagini). După fiecare operație de convoluție, este aplicată funcția ReLU rezultatelor obținute pentru a permite rețelei să învețe tipare mai complexe. [40]

Straturile de grupare (pooling) reduc dimensiunea rezultatului obținut în urma convoluției și minimizează numărul de parametri. La fel ca straturile de convoluție, acestea utilizează filtre glisante, însă diferența este ce aceste filtre nu conțin valorile ponderilor. Există două tipuri principale de grupare a datelor: max pooling, care selctează cea mai mare valoare din fiecare regiune și este cea mai utilizată metodă și average pooling, care calculează o valoare medie. Procesul de pooling oferă beneficii cum ar fi: o generalizare mai bună a rețelei și un risc redus de supraînvățare, dar implică și dezavantaje. Din cauza faptului că este un proces distructiv, o mărime prea mare a filtrului, poate duce la o pierdere excesivă a datelor, astfel reducând considerabil performanțele modelului. [39], [40]

Straturile dense, cunoscute și ca straturi complet conectate, sunt straturi în care fiecare neuron este conectat la toți neuronii din stratul anterior și cel următor. Fiecare neuron aplică o transformare liniară datelor de intrare, urmată de o funcție de activare. Acest tip de conexiune este analog modului în care neuronii sunt aranjați în cadrul rețelelor neuronale artificiale clasice. [39], [41]

### 2.2.3 CoAtNet

CoAtNet reprezintă un model hibrid care combină avantajele rețelelor neuronale convoluționale cu benficiile arhitecturilor de tip Transformer. [42]

Transformer reprezintă o arhitectură de rețele neuronale utilizat pentru sarcini din domeniul procesării limbajului. Acestea sunt bazate pe mecanisumul de „self-attention”, fără a utiliza operații recursive sau convoluționale. Prin acest mecanism, fiecare element dintr-o secvență poate interacționa cu restul elementelor, astfel modelul poate surprinde dependențele dintre acestea prin intermediul produsului scalar. Arhitectura urmează structura „encoder-decoder”, în care „encoderul” generează reprezentări contextuale pentru datele inițiale, în timp ce „decoderul” generează secvența de ieșire. Spre deosebire de metodele clasice, arhitectura Transformer oferă o perspectivă globală asupra datelor, un aspect important în prelucrarea datelor vizuale. [43]

Algoritmul CoAtNet îmbină vertical straturi convoluționale și straturi de atenție, pentru a îmbunătăți capacitatea și eficiența modelului. Arhitectura prezintă blocuri de tip MBConv, care utilizează convoluția „depthwise”, și are o structură organizată în 5 etape, în care rezoluția spațială este redusă treptat. Prima etapă constă într-o operație de convoluție formată din 2 straturi, ulterior urmând etapa în care sunt aplicate blocurile MBConv, necesare din cauza dimensiunii mari a datelor, care ar face ineficientă aplicarea mecanismului de atenției. Începând cu următoarea etapă, se pot implementa atât blocuri convoluționale, cât și cele de tip Transformer, în funcție de cerințele arhitecturii. Această organizare se datorează faptului că straturile de convoluție inițiale sunt potrivite pentru detectarea trăsăturilor de bază, în timp ce în etapele superioare, straturile de atenție se pot concentra asupra dependențelor globale. [42]

Blocul MBConv, a fost prima oară introdus în arhitectura MobileNetV2, și este un modul din cadrul rețelelor neuronale creat pentru a reduce calculele de procesare fără a compromite performanța. Structura sa include un strat de expansiune care mărește dimensiunea datelor, urmat de o convoluție „depthwise”. Spre deosebire de metodele clasice de convoluție care îmbină informațiile din cadrul tuturor canalelor, convoluția „depthwise” procesează datele prin aplicarea unui filtru pentru fiecare canal individual. Ulterior, este utilizat un strat de proiecție pentru a readuce datele la dimensiunea inițială. Dacă dimensiunile datelor de intrare și celor de ieșire corespund, se aplică o conexiune reziduală pentru a facilita o antrenare mai stabilă și mai eficientă. [44]

Totuși, algoritmul CoAtNet prezintă și câteva limitări. Aplicarea directă a mecanismului de atenție pe imagini cu o rezoluție înaltă poate fi costisitor din punct de vedere al calculelor de procesare, motiv pentru care este necesară reducerea dimensiunii datelor. Acest lucru însă, poate provoca pierderi considerabile de informații, în special la nivelul trăsăturilor de bază, fapt ce poate influența negativ performanța algoritmului. De asemenea, arhitectura de tip Transformer necesită cantități mari de date pentru o antrenare corespunzătoare. În contextul antrenării cu un set de date limitat, există un risc major de apariție a fenomenului de supraînvățare, din cauza lipsei bias-urilor specifice rețelelor convoluționale.[42]

### 2.2.4 Procesarea limbajului natural

Procesarea limbajului natural (NLP) reprezintă un domeniu care oferă computerelor abilitatea de a înțelege și de a interacționa folosind limbaj uman. Prin faptul că integrează atât concepte de lingvistică, cât și concepte precum metode statistice sau modele bazate pe ML și DL, NLP permite sistemelor digitale să genereze, să recunoască și să înțeleagă limbaj scris sau vorbit. Există trei metode principale prin care NLP este abordat, și anume: poate fi bazat pe reguli predefinite, NLP statistic, care utilizează algoritmi de învățare automată și NLP bazat pe rețele neuronale. [45], [46]

Procesarea limbajului natural funcționează urmând o anumită structură care începe prin preprocesarea textului, o etapă care include „tokenization”, mai exact separarea în cuvinte sau propoziții, ștergerea cuvintelor uzuale și eliminarea caracterelor cum ar fi punctuația sau numerele. Procesul se continuă cu etapa de extragere a caracteristicilor, în care textul este convertit în numere pentru a putea fi interpretate. Există tehnici prin care sunt înregistrate importanța și apariția cuvintelor, dar și tehnici care pot surprinde relațiile semantice dintre cuvinte. Ulterior, este realizată analiza textului care presupune interpretarea datelor structurate în vederea obținerii unor informații mai relevante. Ultima etapă se referă la antrenarea modelelor de învățare automată pentru a procesa datele din care să invețe tipare și să genereze predicții. [46]

NLP are diverse utilități și este integrat în mai multe aplicații, printre care se numără: sistemele Siri, Alexa, Cortana, extragerea opiniilor din recenzii sau rețelele de socializare, traducerea textelor, instrumente pentru scrierea și editarea conținutului s.a.m.d. [45]

### 2.2.5 Metrici de evaluare a modelului

Metricile de evaluare reprezintă un element important în procesul de analiză a performanțelor modelelor de învățare automată. Ele determină capacitatea modelului de a face predicții corecte, de a generaliza corespunzător și de a permite comparații cu alte modele. Alegerea metricilor depinde de tipul problemei abordate. În probleme de clasificare, cele mai utilizate metrici sunt acuratețea, precizia, senzitivitatea, specificitatea și scorul f1, fiecare oferind informații specifice despre calitatea predicțiilor. Este important ca evaluarea să fie realizată pe un set de date de test, pe care modelul nu le-a mai văzut, pentru obținerea unor rezultate cât mai realiste și obiective. În plus, metricile de evaluare contribuie semnificativ la optimizarea și ajustarea modelului în cadrul procesului de dezvoltare.

Calcularea metricilor necesită definirea următorilor termeni:

TP (True Positive) = predicție pozitivă, valoare reală pozitivă

TN (True Negative) = predicție negativă, valoare reală negativă

FN (False Negative) = predicție negativă, valoare reală pozitivă

FP (False Positive) = predicție pozitivă, valoare reală negativă

Acuratețea reprezintă procentul predicțiilor corecte și este definită matematic conform relației: [31]

(2. 7)

Un model ideal ar obține o acuratețe de 100%, adică ar realiza exclusiv predicții corecte. Acuratețea este o metrică ce ia în considerare toate elementele din matricea confuziilor și este frecvent utilizată atunci când setul de date este echilibrat, adică fiecare clasă are un număr similar de exemple. Pentru un set de date dezechilibrat, atunci când un rezultat FP sau FN are un impact mai mare decât celălalt, alte tipuri de metrici sunt recomandate.

Matricea confuziilor reprezintă o metodă de vizualizare, mai exact un tabel, care evidențiază în ce măsură predicțiile acestuia coincid cu valorile reale. Aceasta arată, pentru fiecare clasă, câte exemple au fost clasificate corect și cât au fost etichetate greșit. [31]

. În cadrul unei matrici de confuzie, fiecare linie indică clasele reale, în timp ce coloanele reprezintă clasele prezise de model. Cea mai rudimentară versiune a unei matrici de confuzie există în contextul clasificării binare și are structura prezentată în Figura 2.6. În cazul clasificărilor multi-clasă, matricea este extinsă, valorile aflate pe diagonala principală reprezentând predicțiile corecte, valorile TP, pentrru fiecare clasă. În timp ce, restul valorilor pot fi interpretate ca TN, FP sau FN în funcție de clasa analizată în momentul respectiv. În scenariul în care modelul clasifică corect toate exemplele, toate valorile se vor regăsi pe diagonală, iar celelalte vor avea valoarea 0. [47]

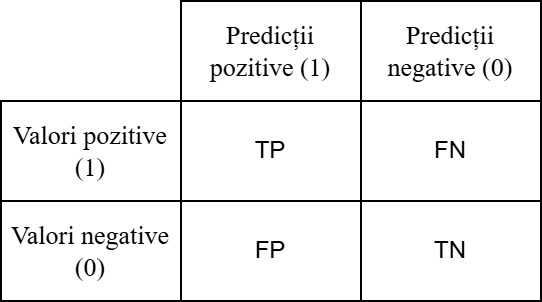


Figura 2. 6 Matricea confuziilor

Precizia reprezintă proporția predicțiilor pozitive care corespund valorilor reale pozitive. Este calculată cu ajutorul formulei:

(2. 8)

Precizia reprezintă o metrică important în situațiile în care rezultatele FP pot avea un impact major. O valoarea ridicată a preciziei indică faptul că predicțiile sunt corecte, minimizând numărul de clasificări eronate. [48]

Senzitivitatea sau rapelul măsoară capacitatea unui model de a detecta corect toate valorile pozitive dintr-un set de date. O valoare ridicată a senzitivității poate duce uneori la o scădere a preciziei, crescând numărul de valori FP. Astfel, cele două metrici sunt adesea utilizate împreună pentru o evaluare echilibrată. Senzitivitatea este calculate conform formulei: [48]

(2. 9)

Între precizie și senzitivitate există un compromis, îmbunătățirea uneia poate duce la degradarea celeilalte. Pentru a gestiona acest dezechilibru, este utilizat scorul F1, care reprezintă media armonică dintre cele două metrici. Scorul F1 acordă o importanță egală, atât senzitivității, cât și preciziei, ceea ce nu este întotdeauna ideal. De aceea, există alte versiuni, cum ar fi scorul F0 sau F2 , care permit ajustarea evaluării în funcție de cerințele aplicației. Scorul F1 este definit astfel: [48]

(2. 10)

Specificitatea reprezintă opusul senzitivității, și indică cât de corect identifică modelul cazurile negative. Aceasta se calculează ca procentul valorilor negative corect clasificate din totalul cazurilor negative (TN și FP). [48]

## 2.4 Tehnologii software utilizate

### 2.4.1 Python

Python este un limbaj de programare de nivel înalt, open-source3, care permite atât programarea procedurală, cât și programarea orientată pe obiecte. Este cunoscut pentru simplitatea sa, sintaxa clară și concisă și pentru biblioteca sa extinsă, ce include numeroase funcții și module incluse. [49]

Fiind un limbaj de nivel înalt, Python abstractizează detalii tehnice legate de hardware, simplificând procesul de scriere. Limbajul oferă o serie de structuri de date predefinite, precum liste, dicționare sau seturi, care pot fi utilizate fără implementări suplimentare De asemenea, include scriere dinamică, ceea ce înseamnă că nu este necesară declararea explicită a variabilelor, acestea sunt determinate automat.. Este un limbaj interpretat, adică codul este executat linie cu linie, ceea ce îl face efficient în detectarea erorilor. [49]

Python este folosit în mai multe domenii, evidențiind versatilitatea sa. Este adesea utilizat în dezvoltarea de aplicații web și software, pentru procesarea imaginilor și conținutului audio, dar în special, în domeniul inteligenței artificiale, pentru care oferă un suport eficient prin numeroase bibllioteci specializate. [49]

### 2.4.2 Pytorch

PyTorch reprezintă un framework4 de învățare profundă care reușește să îmbine flexibilitatea programării dinamice cu performanța necesară algoritmilor de învățare automată. Spre deosebire de framework-urile tradiționale, precum TensorFlow sau Caffe, PyTorch permite utilizatorilor să interacționeze direct cu Python și oferă un mod de funcționare mai versatil. [50]

Un avantaj al PyTorch este că structura modelului poate fi modificată dinamic, în timpul execuției, ceea ce facilitează ajustarea arhitecturii modelului. TensorFlow necesită ca modelul să fie complet definit și compilat înainte de rulare În plus, una dintre funcționalitățile incluse în PyTorch este calculul automat al derivatelor, esențial în procesul de antrenare. [51]

PyTorch obține performanțe ridicate în procesarea calculelor datorită componentei de bază implementată în limbajul C++ și unui sistem de gestionare a memoriei optimizat pentru folosirea mai multor GPU. [50]

Peste 70% din proiectele de cercetare în domeniul AI folosesc PyTorch. [52] Acesta oferă suport pentru numeroase arhitecturi de rețele neuronale, începând cu modele de bază cum ar fi regresia liniară până la rețele convoluționale și modele de tip Transformer.

### 2.4.3 Librosa

Librosa este o bibliotecă Python specializată în analiza semnalelor audio și muzicale. Are ca scop principal, procesarea semnalului audio pentru o analiză eficientă, utilizând o interfață Python. [53]

3 Software al cărui cod sursă este accesibil utilizatorilor pentru a fi viuzalizat sau modificat

4 O structură predefinită care oferă baza pentru dezvoltarea aplicațiilor software

La bază, Librosa permite utilizatorilor să încarce, să manipuleze și să analizeze date audio, prin funcționalități cum ar fi extragerea caracteristicilor, vizualizarea semnalelor sau aplicarea altor transformări audio. Biblioteca oferă instrumente pentru generarea de spectrograme, calculul coeficienților mel-cepstrali sau alte statistici spectrale. Este o bibliotecă construită exclusiv în Python și se integrează eficient cu mediul acestuia, colaborând cu biblioteci precum NumPy sau matplotlib. Unul dintre avantajele prezente în Librosa este că permite personalizarea funcțiilor implicite, un aspect util pentru experimente și cercetare. [53]

Biblioteca Librosa este utilizată în aplicații precum clasificarea genurilor muzicale, sisteme de recomandări muzicale sau preprocesarea audio pentru modele de învățare profundă. [53]

# Capitolul 3. Implementarea practică

## 3.1 Structura proiectului

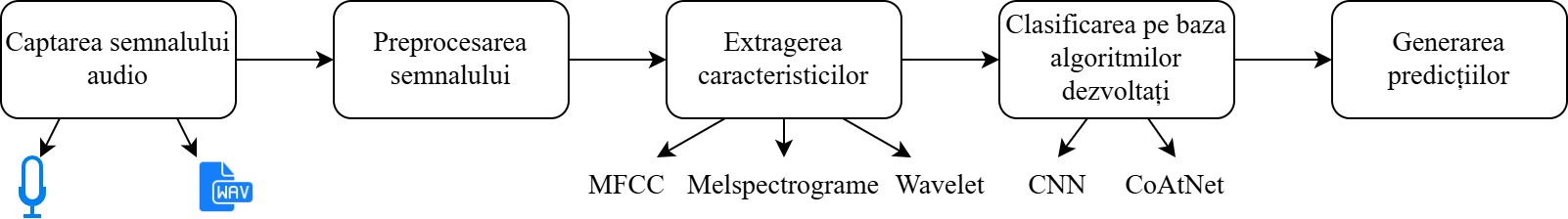


Figura 3. 1 Etapele necesare realizării proiectului

Implementarea practică a proiectului include mai multe etape, precum:

Etapa de captare a semnalului audio reprezintă etapa inițială a procesului și se referă la realizarea înregistrărilor ce conțin sunetele tastelor, care ulterior vor alcătui baza de date. Acest lucru este realizat cu ajutorul unui microfon poziționat la o anumită distanță, care captează semnalele acustice cu o anumită rată de eșantionare și apoi, sunt stocate în format .wav pentru a fi transmise către etapa de preprocesare.

Preprocesarea semnalelor este o etapă importantă ce asigură calitatea și consistența înregistrărilor înainte de a fi utilizate pentru extragerea caracteristicilor. Această etapă presupune: reducerea zgomotului prezent în înregistrări prin aplicarea unor filtre sau algoritmi, normalizarea amplitudinilor semnalelor, și, în final, segmentarea înregistrărilor prin detecția fiecărei apăsări de taste. Normalizarea este un pas care asigură robustețea în cadrul procesului de extragere a trăsăturilor, prin ajustarea amplitudinilor unei forme de undă audio cu scopul de a facilita un nivel standard al acestora, astfel asigurând mai puține variații în cadrul înregistrărilor.

Extragerea caracteristicilor reprezintă o etapă fundamentală în cadrul proiectului, deoarece ea oferă datele necesare pentru algoritmi în vederea clasificării semnalelor. Etapa prespune utilizarea unor reprezentări vizuale ale semnalelor precum Melspectrograme sau bazate pe exragerea Coeficienților Mel-Cepstrali și, respectiv utilizarea Transformatei Wavelet. Fiecare astfel de reprezentare oferă detalii diferite cu privire la proprietățile semnalului.

Următoarea etapă se referă la antrenarea algoritmilor cu ajutorul imaginilor obținute în urma pasului precedent. Fiecare imagine corespunde unei apăsări a tastei și este etichetată corespunzător pentru a permite algoritmilor să învețe tiparele care diferențiază tastele. Scopul acestei etape este de a obține modele capabile de a generaliza, astfel încât să poată clasa semnale noi, ce nu au fost întâlnite în timpul antrenării.

În urma clasificării datelor, se pot obține predicții pentru fiecare semnal acustic segmentat, care formează ulterior secvențe de caractere ce reprezintă datele originale dactilografiate. În cadrul secvențelor, pot exista adesea caractere greșite, din acest motiv este aplicată o tehnică de postprocesare, bazată pe NLP, care sugerează cuvinte alternative, folosind probabilitățile corespondente fiecărei taste obținute cu ajutorul modelului. Acest pas are ca scop reconstruirea unui text în urma procesului de tastare.

Proiectul este structurat în 4 fișiere, fiecare reprezentând o etapă fundamentală din cadrul procesului, astfel: preprocessing.py (Anexa 1), cnn.py (Anexa 2), coatnet.py (Anexa 3) și predict.py (Anexa 4)

## 3.2 Descrierea setului de date

Baza de date utilizată conține aproximativ 56 de înregistrări pentru fiecare tastă, dar pot exista mici variații datorită procesului de înregistrare manuală. Aceste înregistrări reprezintă baza pentru generarea reprezentărilor vizuale. Ele au fost efectuate într-un mediu fără zgomot, cu ajutorul unui smartphone, mai exact Iphone 11, situat la o distanță fixă față de tastatură pentru a păstra uniformitatea și așezat pe un material de microfibră pentru a reduce vibrațiile. Rata de eșantionare a înregistrărilor este de 48000 Hz și acestea conțin mai multe apăsări ale unei taste. Pentru a separa fiecare apăsare, este implementat un algoritm care funcționează pe baza pragurilor energetice. Cu alte cuvinte, o apăsare a tastei este luată în considerare atunci când algoritmul detectează o creștere la nivel energetic în cadrul formei de undă.

Tastatura folosită pentru colectarea datelor este modelul Dark Alien K870 RGB, o tastatură mecanică care are tastele dispuse conform standardului QWERTY. O caracteristică definitorie a tastaturii este prezența comutatoarelor albastre, care oferă un sunet specific. A fost utilizat acest model de tastatură, deoarece față de alte tipuri de tastaturi care pot fi mai silențioase, aceasta permite înregistrarea unor sunete mai clare și mai ușor de recunoscut.

Fișierele rezultate în urma segmentării sunt stocate în format .wav și sunt dispuse în directoare organizate în funcție de clase. Clasele se referă la caracterele incluse în baza de date, care în cadrul acestui proiect se limitează doar la alfabetul englez alcătuit din 26 de litere.

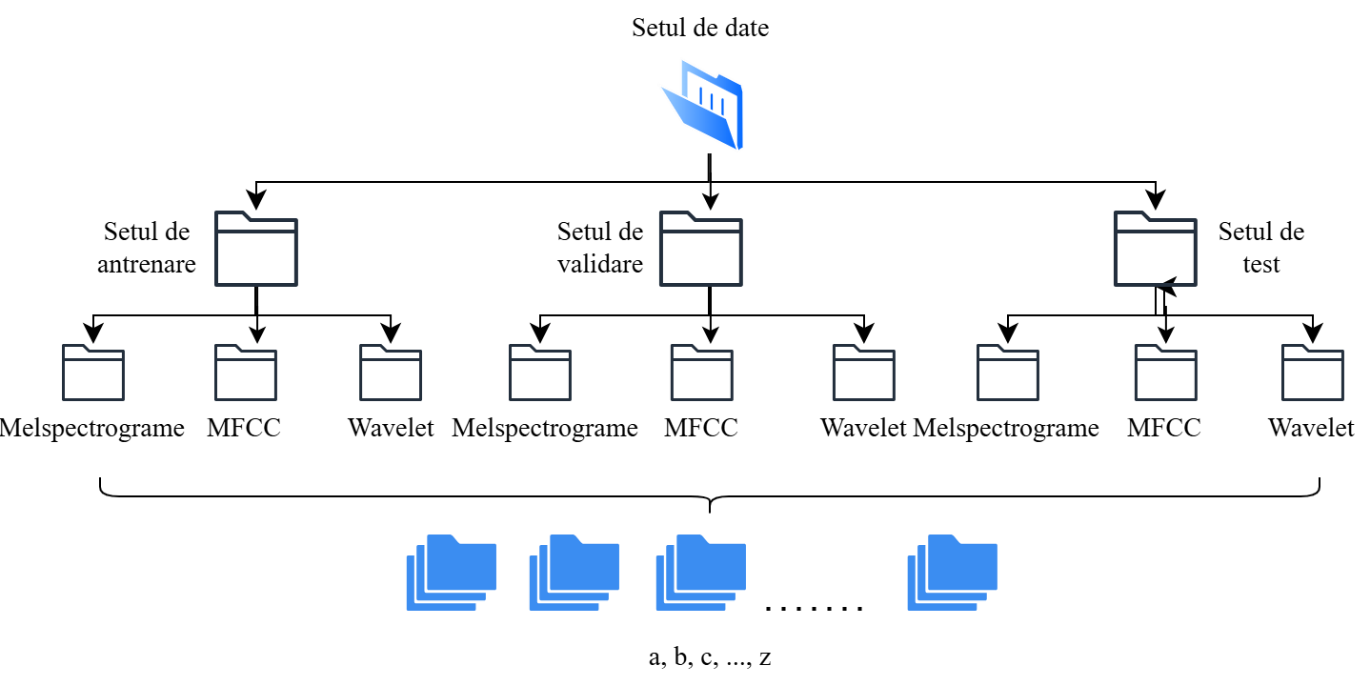


Figura 3. 2 Structura setului de date

Ulterior, fișierele audio segmentate care conțin câte o apăsare a unei taste sunt utilizate pentru a realiza imaginile ce vor alcătui seturile de date folosite în procesul de învățare. Sunt generate, în mod egal pentru fiecare clasă, 1448 de imagini pentru fiecare tip de reprezentare vizuală, astfel asigurând un număr consistent de exemple pentru a menține setul de date echilibrat. Asemenea înregistrărilor segmentate, imaginile sunt stocate sub aceeași formă, în directoare etichetate cu numele claselor corespunzătoare pentru a facilita procesul de învățare supervizată.

Structura setului de date final utilizată în procesul de antrenare este prezentată în figura 3.2. Inițial setul este împărțit în subseturi necesare etapelor incluse în procesul de învățare, ulterior acestea fiind separate în directoarele specifice tipului de reprezentare și respectiv clasei.

## 3.3 Analiza semnalelor audio

Înainte de a putea începe etapa de extragere a caracteristicilor, este esențial să existe o analiză asupra semnalelor brute pentru a observa caracteristici și diferențe între sunetele tastelor. Pentru a obține o perspectivă globală, în această secțiune se va realiza o analiză atât în domeniul timp, cât și în domeniul frecvență. O mențiune importantă este că semnalele emise în timpul apăsării tastelor pot varia considerabil în funcție de stilul de dactilografiere, forța de apăsare a tastelor, dar și de modelul tastaturii, fapt ce reduce robustețea etapei de procesare a datelor.

Înregistrările sunt importate în cadrul proiectului cu ajutorul funcției *load* din cadrul bibliotecii *librosa,* fără a schimba rata de eșantionare standard a microfonului iPhone, lucru care menține precizia temporală. Conform teoremei Nyquist, este important ca frecvența de eșantionare să fie de cel puțin două ori mai mare decât frecvența maximă a semnalului. Rata de eșantionare de 48000 Hz permite ilustrarea frecvențelor de până la 24000 Hz pentru a exista siguranța că nu se pierd detalii esențiale ale semnalelor.

După importarea acestora, semnalul audio este normalizat împărțind fiecare eșantion al semnalului la valoarea maximă absolută a amplitudinii, astfel forma de undă menținându-se în intervalul standard [-1, 1]. Urmează aplicarea funcției *reduce\_noise* din modulul *noisereduce*, care analizează semnalul și elimină zgomotul prezent pentru a genera o versiune mai calitativă a semnalului. Având în vedere faptul că înregstrările conțin mai multe apăsări ale unei taste, este necesară implementarea unui algoritm pentru a realiza segmentarea acestora.

Scopul algoritmului implementat este de a obține separat fiecare apăsare a tastei pentru a fi salvate în fișiere .wav de o lungime fixă. Este utilizată funcția *onset.detect* pentru a identifica momentele de timp în care există o creștere bruscă în cadrul energiei. Acest lucru poate fi subiectiv, deoarece procesul de identificare a momentelor este puternic influențat de parametrii funcției. Parametrul *delta* este considerat un prag, o valoare mai scăzută poate duce la detectarea unor valori fals pozitive, iar o valoare mai mare face procesul de detecție unul mai strict. Parametrii *pre\_max* și *post\_max* se referă la fereastra în care poate fi detectată o apăsare a tastei. În cadrul lucrării, parametrii au fost setați astfel: *delta* = 0.2, *pre\_max* = 10 și *post\_max* = 10. Pentru a nu permite detecția aceleași apăsări de mai multe ori, a fost adăugat un pas care să impună o distanță de 0,3 s între două apăsări succesive. De asemenea, durata unei înregistrări este de 0,3 s, pentru a asigura uniformitatea setului de date. Ultimul pas este reprezentat de salvarea înregistrărilor segmentate în format .wav pe care le stochează în directoare denumite în funcție de clasele existente.

Cum principalul factor care influențează sunetele tastelor este poziția acestora pe placa tastaturii, vom analiza formele de undă pentru tastele *m* și *q*. Tasta *q* este situată în colțul din stânga sus, iar tasta *m* este poziționată în colțul din dreapta jos.

Din figura 3.2, se poate observa că momentul apăsării tastei *q* este caracterizat de o amplitudine mare, apropiată de valoarea maximă normalizată, ceea ce indică un sunet mai puternic față de sunetul apăsării tastei *m*. De asemenea, figura 3.3 ilustrează o distribuție energetică care scade gradual, fapt ce indică prezența oscilațiilor mecanice. În schimb, distribuția energiei pentru tasta *q* se concentrează în primele 80 ms și scade brusc ulterior. O ultimă diferență este reprezentată de faptul că, în cadrul primei figuri, există din nou a creștere bruscă în ampltidine, mai exact în momentul eliberării tastei, lucru care este mai puțin prezent în cazul tastei *m*.

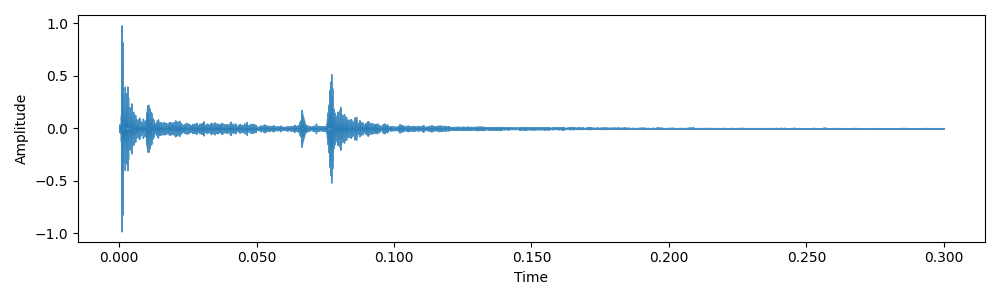


Figura 3. 3 Forma de undă pentru tasta „q”

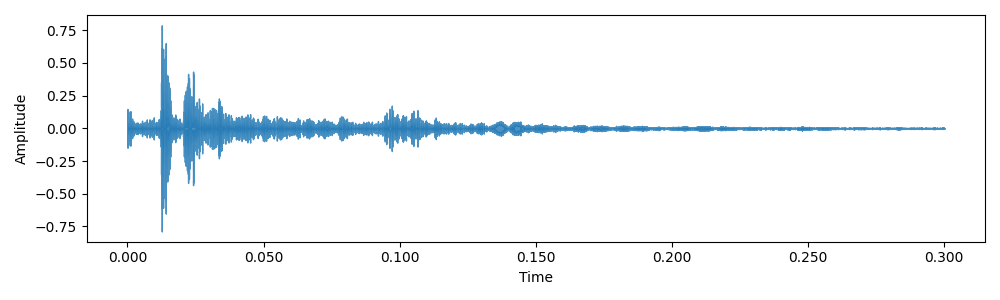


Figura 3. 4 Forma de undă pentru tasta „m”

Reprezentările în timp permit vizualizarea structurii generală a semnalelor, cum ar fi durata și amplitudinea. Însă, nu oferă cea mai bună bază pentru a crea un set de date optim pentru procesul de învățare, deoarece este limitat din punct de vedere al caracteristicilor vizuale care să diferențieze tastele între ele. În schimb, reprezentările frecvențiale pot furniza un set de date mai relevant pentru acest tip de semnale acustice.

Pentru a trece în domeniul spectral, am aplicat funcția *fft* din biblioteca *numpy* care creează un vector de numere complexe ce reprezintă componentele de frecvență ale semalului. Deoarece este dorită o reprezentare bidimensională a spectrului, este importantă utilizarea funcției *abs* pentru valorile complexe obținute, deoarece aceasta calculează valorile reale ale amplitudinilor fiecărei frecvențe.

Figura 3.4 prezintă o componentă spectrală proeminentă în jurul frecvenței de 7000 Hz, lucru care sugerează că tasta *q* produce un sunet cu frecvența relativ ridicată. Acest fapt se poate datora fie aplicării unei forțe mai mari în momentul apăsării, fie poziționării tastei. În figură se poate observa faptul că și restul componentelor sunt destul de pronunțate și nu depășesc frecvența de 10000 Hz. Tasta *m*, în schimb, are un spectru mai larg de frecvențe, cu o componentă dominantă în cadrul frecvențelor joase. Corespondent reprezentării în domeniul timp, distribuția energetică a frecvențelor indică un sunet mai slab și rezonant

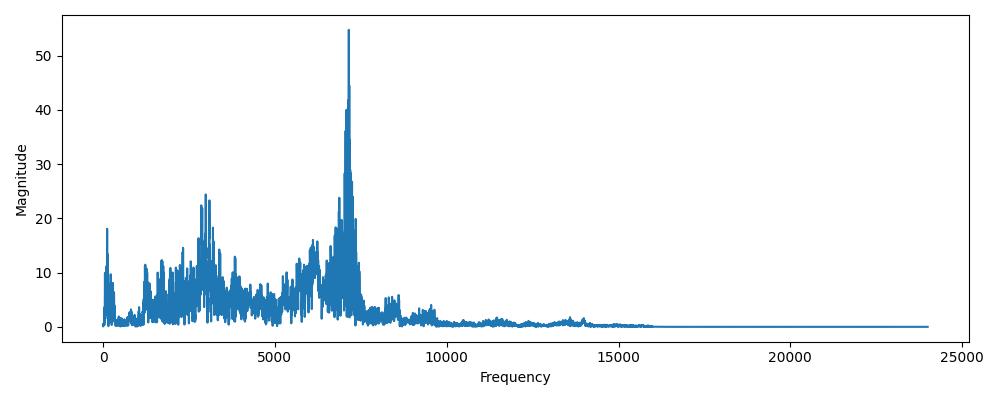


Figura 3. 5 Analiza în domeniul frecvență pentru tasta „q”

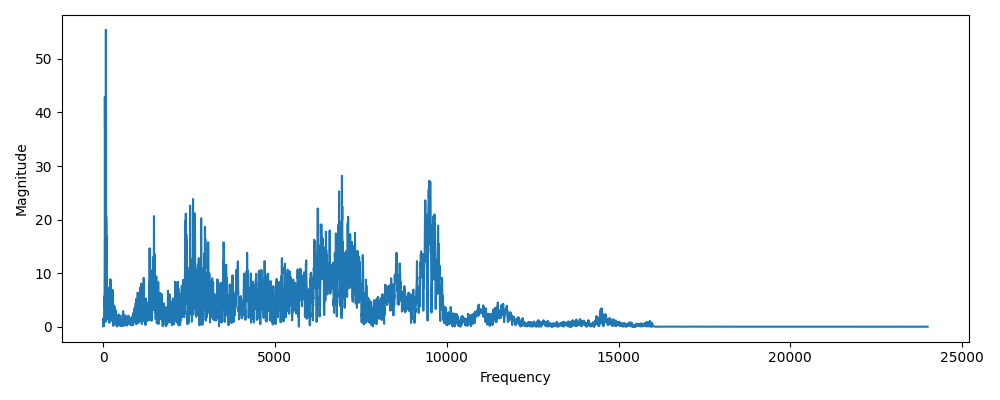


Figura 3. 6 Analiza în domeniul frecvență pentru tasta „m”

Comparațiile anterioare demonstrează diferențele ce pot exista între sunetele tastelor și influența factorilor asupra acestora. Tasta *q* a generat un sunet mai intens, cu energie concentrată în zona frecvențelor mai înalte, în timp ce tasta *m* a produs un sunet caracterizat frecvențe mai joase, fiind mai rezonant

## 3.4 Extragerea caracteristicilor

Modelele de învățare automată necesită un set de date care să poată furniza informații mai ofertante din punct de vedere a trăsăturilor unui semnal acustic. De aceea, procesul de extragere al caracteristicilor este unul esențial în cadrul proiectului.

Melspectrogramele au fost generate cu ajutorul funcției *melspectrogram* oferită de modulul *librosa*. Această funcție creează un vector 2D, în care fiecare valoare este reprezentată de energia unei benzi mel la un anumit moment de timp. Parametrii *n\_fft* și *hop\_length* ai funcției, setați cu valorile 2048 și respectiv 512 determină rezoluția Transformatei Fourier pe Termen Scurt, controlând modul în care semnalul este împărțit în cadre. Parametrul *n\_mels* specifică în câte benzi mel este împărțită axa frecvențelor, în acest caz fiind 128. Melspectrograma este convertită dintr-o scară liniară într-una logaritmică, exprimată în dB, folosind funcția *power.to.dB*, pentru a crea o reprezentare vizuală mai potrivită clasificării semnalelor.

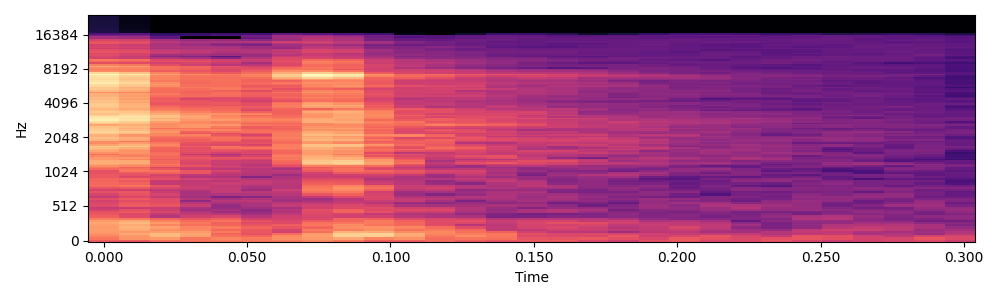


Figura 3. 7 Melspectrograma pentru tasta „q”

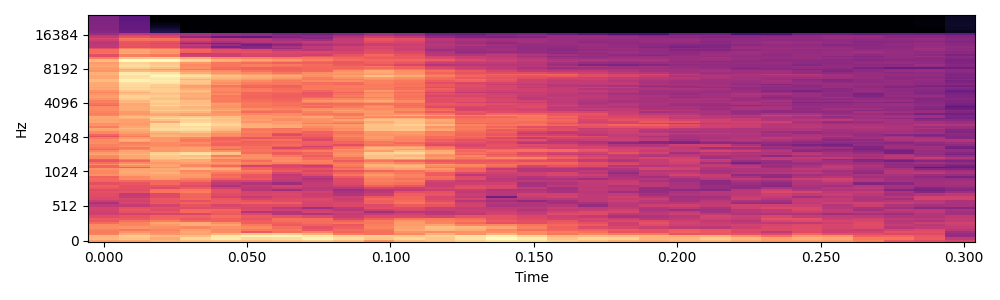


Figura 3. 8 Melspectrograma pentru tasta „m”

Pentru a continua analiza semnalelor și din alte perspective, vom utiliza tot exemple pentru tastele *q* și *m,* pentru care s-a demonstrat anterior că arată diferențe structurale și în analiza temporală și în cea spectrală. Examinând și reprezentări precum Melspectrograme, MFCC sau coeficienți obținuți cu ajutorul Transformatei Wavelet se pot observa și alte informații care nu sunt la fel de vizibile în reprezentările anterioare.

În figura 3.6, corespondentă tastei q, se observă o concentrare a energie între 3000 și 8000 Hz, în special în intervalul 0-0,1 ms, care apoi scade, confirmând natura intensă și scurtă a semnalului. Figura 3.7 arată că energia este predominant localizată între 500 și 4000 Hz, aceasta fiind mai uniform distribuită și scade treptat, la fel ca în reprezentarea spectrală a tastei *m*. Melspectrogramele sunt reprezentări timp-frecvență, permițănd astfel ilustrarea ambelor caracteristici.

Coeficienții Mel-Cepstrali au fost extrași din semnalele audio prin intermediul funcției *mfcc* din cadrul pachetul *librosa*, care utilizează STFT cu dimensiunea unei ferestre de 2048 de eșantioane și distanța între ferestre de 512 de eșantioane. Pentru o reprezentare mai adecvată, este eliminat primul coeficient extras, deoarece aceasta prezintă o imagine globală a energiei semnalului și nu contribuie la diferențierea claselor, în timp ce restul coeficienților se axează pe informații mai relevante clasificării semnalelor acustice.

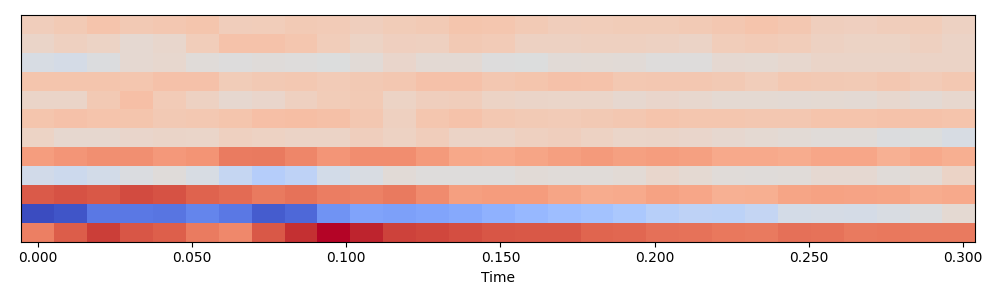


Figura 3. 9 Reprezentarea MFCC a tastei „q”

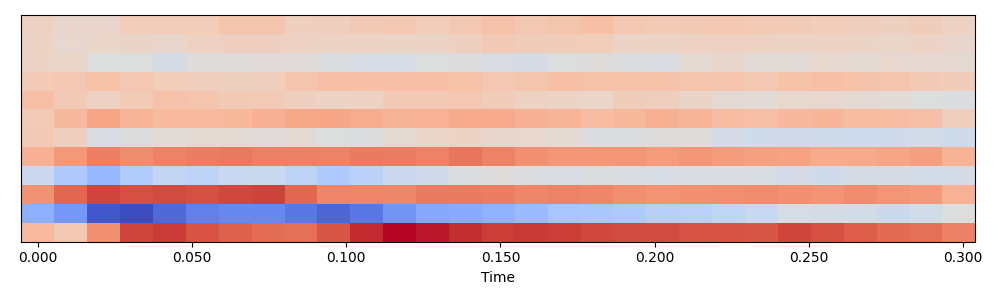


Figura 3. 10 Reprezentarea MFCC a tastei „m”

În cadrul unei reprezentări a Coeficienților Mel-Cepstrali, axa Y reprezintă coeficienții, care ilustrează caracteristici diferite ale semnalelor. Primii doi coeficienți evidențiază diferențe dintre frecvențele înalte și cele joase pentru a detecta intensitatea generală a semnalului. Coeficienții 3-5 diferențiază între semnale cu un caracter brusc de semnalele cu un caracter mai rezonant, iar coeficienții 6-8 ilustrează detalii mai fine care influențează spectrul, cum ar fi construcția tastaturii sau poziționarea microfonului. Ultimii coefiecienți prezintă variații subtile sau componente de zgomot prezente în cadrul semnalului. Cu alte cuvinte, primii coeficienți se axează pe caracteristici generale, în timp ce coeficienții superior pun accent pe detaliile sunetului.

Figurile 3.8 și 3.9 confirmă diferențele dintre tastele q și m obținute anterior, însă adaugă și alte informații utile. Un aspect suplimentar al reprezentării MFCC este abilitatea de a evidenția evoluția profilului spectral în timp. Pentru tasta *q*, coeficienții arată variații mai abrupte, indicând faptul că profilul spectral al semnalului se schimbă rapid. Tasta *m* prezintă mai puține fluctuații ceea ce sugerează un semnal mai uniform și stabil. În plus, se poate observa că în cazul reprezentării pentru tasta q sunt activați mai mulți coeficienți în primele cadre, fiind un semnal cu o structură armonică mai complexă, spre deosebire de sunetul tastei m care arată un număr mai redus de coeficienți activați.

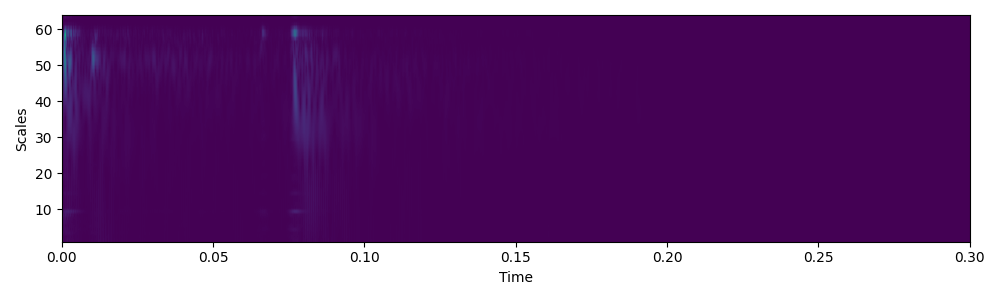


Figura 3. 11 Reprezentarea Wavelet pentru tasta „q”

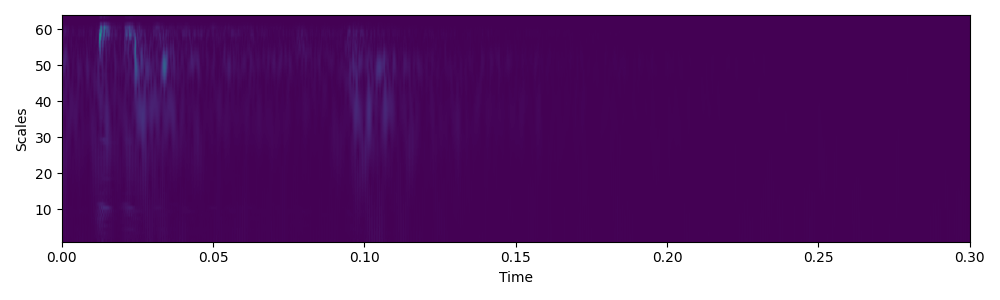


Figura 3. 12 Reprezentarea Wavelet pentru tasta „m”

Pentru a putea calcula coeficienții Transformatei Wavelet, a fost definită funcția *wavelet\_transform.* În cadrul funcției, este folosită funcția predefinită *cwt* din utilitarul *pywt*, care are ca parametrii tipul funcției wavelet utilizat, în cazul proiectului fiind morlet, și o valoare pentru a determina rezoluția analizei wavelet. Funcția returnează o matrice alcătuită din valori complexe ale coeficienților, care ulterior sunt afișați sub forma unei imagini.

Reprezentările Wavelet oferă mai multe detalii din punct de vedere temporal, astfel evidențiind semnale tranzitorii. Figura 3.10 prezintă impulsuri de energie izolate, lucru care se putea observa și în reprezentări anterioare, dar cu o precizie temporală mai mică. Pentru tasta *m*, figura 3.11 descrie o reprezentare mai continuă, corespondent tranzițiilor mai fluide.

Inițial toate reprezentările realizate sunt de tip RGBA5, din acest motiv pentru a asigura compatibilitatea dintre acestea și modelele dezvoltate este necesară convertirea acestora în modul RGB. Convertirea imaginilor este realizată cu ajutorul funcției *convert* din cadrul modului *Imagine* importat din biblioteca PIL. Ulterior sunt salvate în format .png și sunt stocate corespunzător organizării din figura 3.2.

## 3.5 Arhitecturi și parametri

### 3.5.1 CNN

Rețelele neuronale convoluționale reprezintă una dintre cele mai utilizate metode de clasificare a imaginilor datorită capacității lor de a identifica tipare și caracteristici în date bidimensionale. Din acest motiv, straturile utilizate în cadrul rețelei sunt de tip 2D.

Rețeaua conține 33 de straturi și este structurată astfel:

* 8 straturi de grupare
* 8 straturi de dezactivare a neuronilor6
* 7 straturi convoluționale
* 7 straturi de normalizare7
* 2 straturi dense
* 1 strat de aliniere8

5 Un model de bază utilizat pentru reprezentarea și afișarea culorilor, în care fiecare literă se referă la un canal separate (Red, Green, Blue), uneori adăugându-se și litera A care indică transparența imaginii

6Dezactivează un anumit procent de neuroni pentru a preveni fenomentul de supraînvățare

7Ajustează valorile de intrare pentru a se încadra în anumite intervale

8Transformă datele în format unidimensional

Figura 3.10 nu prezintă doar structura rețelei, ci și evoluția dimensiunii datelor în urma trecerii prin straturile rețelei. Se observă o micșorare graduală a rezoluției imaginilor, un aspect care crește eficiența modelului. În același timp, numărul de canale crește, astfel permițând straturilor superioare să învețe caracteristicile definitorii pentru realizarea diferențierii între taste. Pentru a menține claritatea figurii și a nu adăuga detalii excesive, funcțiile de activare ReLU au fost omise, dar acestea sunt aplicate de-a lungul întregii arhitecturi, după fiecare strat de normalizare și primul strat dens. Ulterior, datele sunt transformate în format unidimensional pentru a putea fi transmis mai departe ultimului strat, și anumel cel de clasificare.

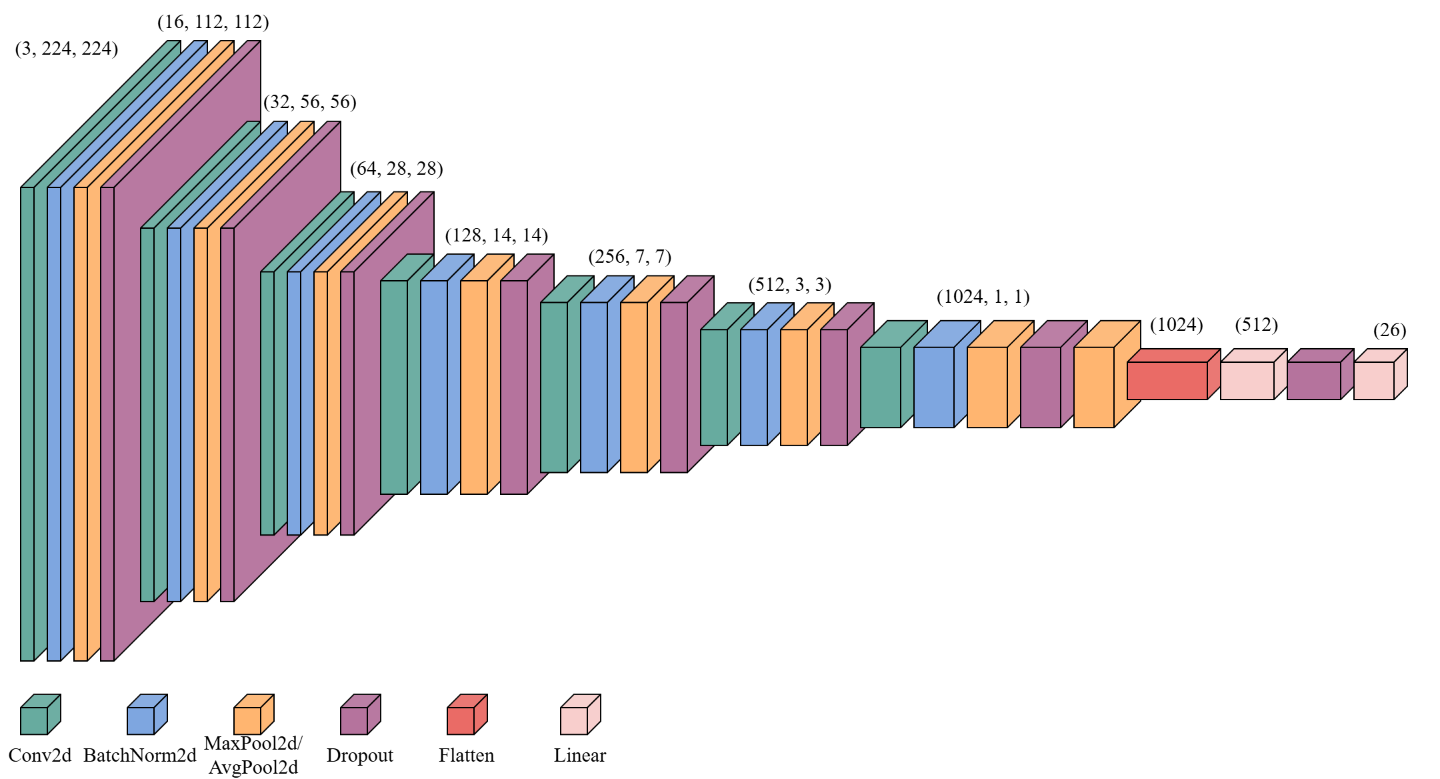


Figura 3. 13 Structura CNN

Fiecare strat are rolul său pe care îl îndeplinește cu ajutorul parametrilor care îl definesc. Prin ajustarea parametrilor, aspecte precum capacitatea de învățare, abilitatea de a generaliza și eficiența de calcul ale rețelei sunt puternic influențate. Parametrii sunt configurați astfel, cu mențiunea că cei neprecizați sunt setați cu valorile standard predefinite de Pytorch:

* Conv2d: *in\_features* = 3, 16, 32, 64, 128, 256, 512, *out\_features* = 16, 32, 64, 128, 256, 512, 1024, *kernel\_size* = 3, *stride* = 1, *padding* =1
* BatchNorm2d: *num\_features* = 16, 32, 64, 128, 256, 512, 1024
* MaxPool2d: *kernel\_size* = 2
* Dropout: *p* = 0,1, 0,2, 0,3, 0,4, 0,5
* Linear: *in\_features*=1024, 512, *out\_features* = 512, 26

### 3.5.2 CoAtNet

Combinând două tehnologii, CoAtNet este cunoscut pentu abilitățile sale de a identifica tipare și caracteristici în imagini complexe. Însă, pentru procesul lor de învățare, Transformers necesită un set de date foarte mare pentru a putea evita apariția supraînvățării. În plus, mecanismul „self-attention” implică faptul că pe măsură ce dimensiunea datelor de intrare se mărește, nevoia de memorie și puterea de calcul se mărește considerabil. Din acest motiv, pentru a asigura condițiile optime, sunt necesare echipamente hardware performante.

Prin urmare, pentru a preveni apariția acestor probleme, algoritmul CoAtNet implementat în cadrul proiectului presupune utilizarea doar blocurilor convoluționale. Algoritmul este open-source, la care am adus următoarele contribuții: înlocuirea blocurilor de tip Transformers cu blocuri de tip MBConv și reducerea numărului de blocuri prezente în fiecare stagiu cu scopul de a reduce complexitatea sistemului fără a pierde din performanța modelului.

|  |  |  |
| --- | --- | --- |
| Stagiu | Mărime | CoAtNet-0 |
| S0 – Conv | 1/2 | L=2 D=64 |
| S1 – MBConv | 1/4 | L=2 D=96 |
| S2 – MBConv | 1/8 | L=2 D=192 |
| S3 – MBConv | 1/16 | L=2 D=384 |
| S4 – MBConv | 1/32 | L=1 D=768 |

Tabel 1 Configurația arhitecturală a modelului CoAtNet-0

Configurația descrisă în Tabelul 1 presupune caracteristicile fiecărui stagiu al algoritmului, în acest caz fiind vorba doar de stagii convoluționale. Pentru fiecare stagiu, există o valoare care indică cu cât se reduc dimensiunile datelor și un anumit număr de blocuri repetate (L) și de canale de ieșire (D). CoAtNet-0 este cel mai de bază model inclus în familia modelelor CoAtNet, care demonstrează influența ajustării valorilor L și D.

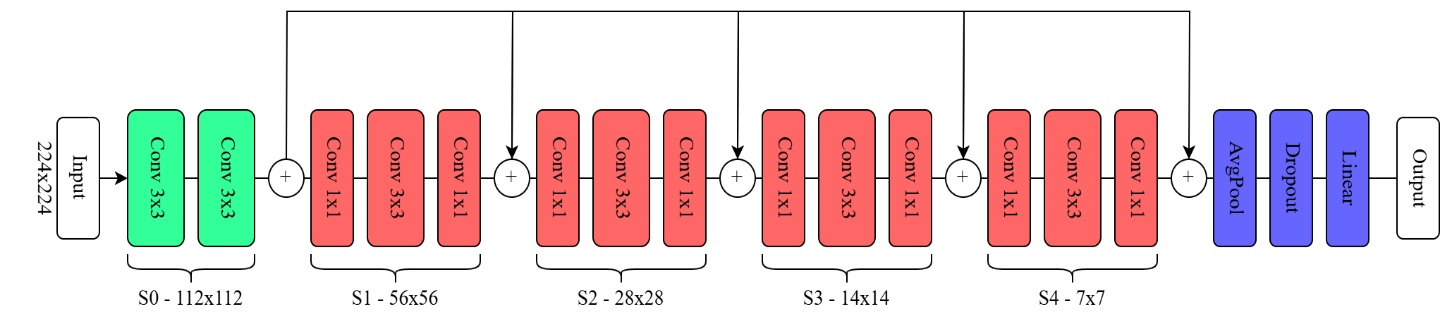


Figura 3. 14 Structura utilizată pentru modelul CoAtNet

În figura 3.14 sunt prezentate stagiile algoritmului și ce implică acestea. În cadrul fiecărui bloc, există atât straturi de normalizare, de grupare și dezactivare a neuronilor, cât și funcții de activare pentru optimizarea modelului. De asemenea, fiecare stagiu include și conexiuni reziduale, fapt ce permite datelor de intrare să fie adăugate rezultatului ulterior.

Parametrii modelului sunt definiți într-un mod mai complex, deoarece algoritmul implică mai multe procese în cadrul arhitecturii sale. Primul bloc convoluțional implică o structură caracterizată de următorii parametri:

* Conv2d*: kernel\_size* = 3, *stride* = 1, 2, *padding* = 1, *bias* = False
* BatchNorm2d
* GELU

Pentru blocurile MBConv, sunt setați parametrii expansion = 4 și downsample – False, înafară de primul bloc din fiecare stagiu, ceea ce înseamnă că este utilizată următoarea configurație:

* Conv2d: *kernel\_size* = 1, *stride* = 1, 2, *padding* = 0, *bias* = False
* BatchNorm2d
* GELU
* Conv2d: *kernel\_size* = 3, *stride* = 1, *padding* =1, *bias* = False (este setat și parametrul *groups* care este egal cu numărul de canal intermediare, acesta fiind stratul de convoluție „depthwise”
* BatchNorm2d
* GELU
* Dropout: p = 0,2
* Conv2d: *kernel\_size* = 1, *stride* =1, *padding* = 0, *bias* = False

Primul strat convoluțional reprezintă stratul de expansiune care mărește numărul de canale pentru a permite învățarea unor caracteristici complexe fără a crește semnificativ numărul de parametri. Stratul de proiecție, ultimul strat, are rolul de a reduce numărul de canale, astfel comprimând informația învățată în straturile de convoluție precedente.

Atunci când parametrul downsample = True, există câteva schimbări structurale:

* MaxPool2d: *kernel\_size* = 3, *stride* = 2, *padding* = 1
* Conv2d: *kernel\_size* = 1, *stride* = 1, *padding* = 0, *bias* = False

Aceste modificări sunt utilizate pentru a asigura tranziția dintre rezoluții și a implementa o conexiune reziduală care să fie compatibilă din punct de vedere al dimensiunilor.

Dupa finalizarea blocurilor MBConv, există trei straturi finale:

* AdapativeAvgPool2d: *output\_size* = (1, 1)
* Dropout: *p* = 0,5
* Linear: *in\_features* = 768, *out\_features* = 26, *bias* = False

Numărul canalelor de intrare, de ieșire și cele intermediare nu sunt setate manual pentru fiecare start, ci sunt definite dinamic în funcție de o listă predifinită ale cărei valori se găsesc în tabelul 1. Există o singură excepție în cadrul blocurilor MBConv, în care un număr de canale intermediare este calculat prin multiplicarea numărului canalelor de intrare cu factorul de expansiune.

## 3.6 Etapa de antrenare

Procesul de învățare include trei etape principale: etapa de antrenare, de validare și cea de test. În timpul antrenării, modelul învață prin ajustarea ponderilor sistemului, folosind algoritmii descriși în capitolul 2.2.1. Etapa de validare este realizată la sfârșitul fiecărei epoci de antrenare, diferența fiind că aceasta nu include modificarea parametrilor rețelei. Pentru a obține un model performant, valorile rezultate în urma etapei de validare trebuie să fie cât mai apropiate de valorile obținute după finalizarea antrenării. Scopul etapei de validare este de a evalua performanța modelului pentru a putea ajusta hiperparametrii. Pentru a putea realiza comparații între diferite scenarii, antrenarea s-a efectuat în aproximativ aceleași condiții. Scenariile comparate sunt:

* Antrenarea ambelor modele cu același set de date
* Antrenarea aceluiași model cu seturi de date diferite

Setul de date este împărțit în funcție de cele trei etape, respectând următoarele proporții: 70% pentru antrenare, 15% pentru validare și 15% pentru testare. Pentru procesarea setului de date și împărțirea acestuia a fost implentată funcția *split*. Inițial, imaginile sunt redimensionate la valorile 224x224 pentru a uniformiza datele de intrare, iar apoi sunt convertite în tensori, folosind modulul *transforms* oferit de biblioteca *torchvision*, care scalează valorile pixelilor în intervalul [0, 1]. Un pas important în procesarea setului de date este reprezentat de folosirea utilitarului *ImageFolder* din modulul *datasets*, care organizează și etichetează imaginile în funcție de structura directoarelor. Ulterior, este utilizată funcția *random\_split* pentru a împărți setul de date în mod aleatoriu și cu o distribuție echilibrată a claselor. În final, utilitarul *DataLoader* asigură încărcarea datelor sub forma unor loturi („batches”) și opțional, poate amesteca datele pentru a evita învățarea mecanică, un aspect necesar în cazul etapei de antrenare. Numărul de loturi este un hiperparametru și se referă la numărul de exemple care sunt procesate în același timp într-un ciclu de ajustare a parametrilor.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numărul de epoci | Funcția de cost | Optimizatorul | Rata de învățare | Numărul de loturi |
| 10/50 | CrossEntropyLoss | Adam | 0,0001 | 8 |

Tabel 2 Parametrii de antrenare

Singurul parametru care diferă în configurațiile de antrenare pentru cele două modele este numărul de epoci. Acest lucru se datorează faptului că algoritmul CoAtNet, spre deosebire de CNN, obține o acuratețe ridicată în mai puține epoci, fapt ce face redundantă setarea unui număr mare de epoci..

Optimizatorul Adam are ca parametrii configurați rata de învățare și „weight\_decay”, setat la valoarea 0,0001. Setarea acestui parametru reprezintă o tehnică de regularizare pentru a evita apariția supraînvățării prin penalizarea valorilor mari ale ponderilor.

În timpul antrenării, sunt măsurați timpii necesari finalizării fiecărei epoci pentru a putea analiza eficiența modelelor, în final calculând durata totală a procesului de antrenare. Acest lucru este realizat folosind funcția *timer*, importată din modulul *timeit*. Timpii de antrenare diferă în funcție de complexitatea arhitecturii modelelor (numărul de straturi, funcții de activare, numărul de neuroni etc.) și de volumul datelor de intrare. În cadrul proiectului, durata procesului de învățare a fost:

* CNN-Melspectrograme:
* CNN-MFCC:
* CNN-Wavelet:
* CoAtNet-Melspectrograme:
* CoAtNet-MFCC:
* CoAtNet-Wavelet:

Durata antrenării în cazul utilizării algoritmului CoAtNet este semnficativ mai mare deoarece DE COMPLETAT

Pentru a evalua calitatea procesului de antrenare, au fost calculate și reprezentate grafic acuratețea și pierderea, atât pentru etapa de antrenare, cât și pentru cea de validare. Metricile au fost reprezentate grafic prin implementarea a două funcții separate, *plot\_loss* și *plot\_accuracy*, utilizând biblioteca *matplotlib*.

## 3.7 Etapa de testare

După finalizarea etapelor de antrenare și validare, urmează etapa de testare care evaluează cât de bine generalizează modelul pentru date complet noi. Această etapă arată că îmbunătățirile apărute în cadrul primei părți a procesului de învățare există și într-un scenariu real, demonstrând astfel performanța sistemului într-o aplicație practică.

În timpul procesului de antrenare, modelul cu cea mai bună acuratețe de validare obținută pentru o anumită epocă este salvat și este apoi utilizat pentru a realiza etapa de test.

# Bibliografie

[1] D. Asonov and R. Agrawal, “Keyboard acoustic emanations,” in *IEEE Symposium on Security and Privacy, 2004. Proceedings. 2004*, Berkeley, CA, USA: IEEE, 2004, pp. 3–11. doi: 10.1109/SECPRI.2004.1301311.

[2] Mubarek Mohammed, “Acoustic Side Channel Attack on Keyboard,” 2024, doi: 10.13140/RG.2.2.29394.98246.

[3] Z. Kh. Abdul and A. K. Al-Talabani, “Mel Frequency Cepstral Coefficient and its Applications: A Review,” *IEEE Access*, vol. 10, pp. 122136–122158, 2022, doi: 10.1109/ACCESS.2022.3223444.

[4] L. Zhuang, F. Zhou, and J. D. Tygar, “Keyboard acoustic emanations revisited,” in *Proceedings of the 12th ACM conference on Computer and communications security*, Alexandria VA USA: ACM, Nov. 2005, pp. 373–382. doi: 10.1145/1102120.1102169.

[5] Y. Berger, A. Wool, and A. Yeredor, “Dictionary attacks using keyboard acoustic emanations,” in *Proceedings of the 13th ACM conference on Computer and communications security*, Alexandria Virginia USA: ACM, Oct. 2006, pp. 245–254. doi: 10.1145/1180405.1180436.

[6] T. Halevi and N. Saxena, “Keyboard acoustic side channel attacks: exploring realistic and security-sensitive scenarios,” *Int. J. Inf. Secur.*, vol. 14, no. 5, pp. 443–456, Oct. 2015, doi: 10.1007/s10207-014-0264-7.

[7] J. Harrison, E. Toreini, and M. Mehrnezhad, “A Practical Deep Learning-Based Acoustic Side Channel Attack on Keyboards,” 2023, doi: 10.48550/ARXIV.2308.01074.

[8] S. A. Anand and N. Saxena, “Keyboard Emanations in Remote Voice Calls: Password Leakage and Noise(less) Masking Defenses,” in *Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy*, Tempe AZ USA: ACM, Mar. 2018, pp. 103–110. doi: 10.1145/3176258.3176341.

[9] A. Compagno, M. Conti, D. Lain, and G. Tsudik, “Don’t Skype &amp; Type! Acoustic Eavesdropping in Voice-Over-IP,” 2016, *arXiv*. doi: 10.48550/ARXIV.1609.09359.

[10] Z. Martinasek, V. Clupek, and K. Trasy, “Acoustic attack on keyboard using spectrogram and neural network,” in *2015 38th International Conference on Telecommunications and Signal Processing (TSP)*, Prague, Czech Republic: IEEE, Jul. 2015, pp. 637–641. doi: 10.1109/TSP.2015.7296341.

[11] S. Turkle, *Alone together: why we expect more from technology and less from each other*. New York: Basic books, 2011.

[12] “How artificial intelligence is transforming the world,” Brookings. Accessed: Apr. 12, 2025. [Online]. Available: https://www.brookings.edu/articles/how-artificial-intelligence-is-transforming-the-world/

[13] I. H. Sarker, “Machine Learning: Algorithms, Real-World Applications and Research Directions,” *SN Comput. Sci.*, vol. 2, no. 3, p. 160, May 2021, doi: 10.1007/s42979-021-00592-x.

[14] D.-T. Mai, D.-C. Nguyen, and T.-N. Pham, “A Keystroke Audio-based Password Prediction Attack using Deep Learning,” in *2024 International Conference on Advanced Technologies for Communications (ATC)*, Ho Chi Minh City, Vietnam: IEEE, Oct. 2024, pp. 774–778. doi: 10.1109/ATC63255.2024.10908124.

[15] P. K., “Acoustic Wave,” in *Acoustic Waves - From Microdevices to Helioseismology*, M. G. Beghi, Ed., InTech, 2011. doi: 10.5772/17765.

[16] K. Lashkari, B. Friedlander, J. Abel, and B. McQuiston, “Classification of transient signals (acoustic signals),” in *ICASSP-88., International Conference on Acoustics, Speech, and Signal Processing*, New York, NY, USA: IEEE, 1988, pp. 2689–2692. doi: 10.1109/ICASSP.1988.197204.

[17] D. K. Staff, “All the Parts of a Mechanical Keyboard Explained,” Das Keyboard Mechanical Keyboard Blog. Accessed: Apr. 05, 2025. [Online]. Available: https://www.daskeyboard.com/blog/parts-of-a-mechanical-keyboard/

[18] M. S. Ahmad, “Deep Learning 101: Lesson 23: The Basics of Audio Signal Processing with FFT,” Medium. Accessed: Apr. 06, 2025. [Online]. Available: https://muneebsa.medium.com/deep-learning-101-lesson-23-the-basics-of-audio-signal-processing-with-fft-ffef65689c1d

[19] “Understanding the Hanning Window: A Practical Guide for Beginners,” Wray Castle. Accessed: Apr. 07, 2025. [Online]. Available: https://wraycastle.com/blogs/knowledge-base/hanning-window

[20] Y. A. Yousef, V. Cosman, and M. Rey, “Time Frequency Analyses of stationary and non-stationary signals using MATLAB functions and Toolboxes),” Jul. 17, 2024, *In Review*. doi: 10.21203/rs.3.rs-4749352/v1.

[21] L. Roberts, “Understanding the Mel Spectrogram,” Analytics Vidhya. Accessed: Apr. 07, 2025. [Online]. Available: https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53

[22] mlearnere, “Learning from Audio: The Mel Scale, Mel Spectrograms, and Mel Frequency Cepstral Coefficients,” TDS Archive. Accessed: Apr. 07, 2025. [Online]. Available: https://medium.com/data-science/learning-from-audio-the-mel-scale-mel-spectrograms-and-mel-frequency-cepstral-coefficients-f5752b6324a8

[23] “Audio Comparison Using MFCC and DTW,” TestDevLab Blog. Accessed: Apr. 09, 2025. [Online]. Available: https://www.testdevlab.com/blog/audio-comparison-using-mfcc-and-dtw

[24] H. Fayek, “Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What’s In-Between,” Haytham Fayek. Accessed: Apr. 09, 2025. [Online]. Available: https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html

[25] G. Strang, “The Discrete Cosine Transform,” *SIAM Rev.*, vol. 41, no. 1, pp. 135–147, Jan. 1999, doi: 10.1137/S0036144598336745.

[26] “Fourier vs. Wavelet Transformations,” Built In. Accessed: Apr. 10, 2025. [Online]. Available: https://builtin.com/data-science/wavelet-transform

[27] T. Konstantinovsky, “Wavelet Transform: A Practical Approach to Time-Frequency Analysis,” The Pythoneers. Accessed: Apr. 10, 2025. [Online]. Available: https://medium.com/pythoneers/wavelet-transform-a-practical-approach-to-time-frequency-analysis-662bdadeb08b

[28] B. Russel and J. Han, “Jean Morlet and the continuous wavelet transform.” CREWES Res., 2016.

[29] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[30] “What is deep learning? | McKinsey.” Accessed: Apr. 13, 2025. [Online]. Available: https://www.mckinsey.com/featured-insights/mckinsey-explainers/what-is-deep-learning

[31] D. M. Dobrea, *Tehnici de inteligenţă computaţională: aplicaţii în electronică şi biomedicină*. Iaşi: Performantica, 2018.

[32] M. A. Nielsen, “Neural Networks and Deep Learning,” 2015, Accessed: Apr. 13, 2025. [Online]. Available: http://neuralnetworksanddeeplearning.com

[33] “Activation Functions in Neural Networks [12 Types & Use Cases].” Accessed: Apr. 13, 2025. [Online]. Available: https://www.v7labs.com/blog/neural-networks-activation-functions

[34] R. Qamar and B. Ali Zardari, “Artificial Neural Networks: An Overview,” *Mesopotamian J. Comput. Sci.*, pp. 130–139, Aug. 2023, doi: 10.58496/MJCSC/2023/015.

[35] “Backpropagation in Neural Network | GeeksforGeeks.” Accessed: Apr. 18, 2025. [Online]. Available: https://www.geeksforgeeks.org/backpropagation-in-neural-network/

[36] “What is Backpropagation? | IBM.” Accessed: Apr. 18, 2025. [Online]. Available: https://www.ibm.com/think/topics/backpropagation

[37] “What is Gradient Descent? | IBM.” Accessed: Apr. 18, 2025. [Online]. Available: https://www.ibm.com/think/topics/gradient-descent

[38] “Recent advances and applications of deep learning methods in materials science | npj Computational Materials.” Accessed: Apr. 18, 2025. [Online]. Available: https://www.nature.com/articles/s41524-022-00734-6

[39] K. O’Shea and R. Nash, “An Introduction to Convolutional Neural Networks,” Dec. 02, 2015, *arXiv*: arXiv:1511.08458. doi: 10.48550/arXiv.1511.08458.

[40] “What are Convolutional Neural Networks? | IBM.” Accessed: Apr. 18, 2025. [Online]. Available: https://www.ibm.com/think/topics/convolutional-neural-networks

[41] “Fully Connected Layer vs. Convolutional Layer: Explained | Built In.” Accessed: Apr. 18, 2025. [Online]. Available: https://builtin.com/machine-learning/fully-connected-layer

[42] Z. Dai, H. Liu, Q. V. Le, and M. Tan, “CoAtNet: Marrying Convolution and Attention for All Data Sizes,” Sep. 15, 2021, *arXiv*: arXiv:2106.04803. doi: 10.48550/arXiv.2106.04803.

[43] A. Vaswani *et al.*, “Attention Is All You Need,” Aug. 02, 2023, *arXiv*: arXiv:1706.03762. doi: 10.48550/arXiv.1706.03762.

[44] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “MobileNetV2: Inverted Residuals and Linear Bottlenecks,” Mar. 21, 2019, *arXiv*: arXiv:1801.04381. doi: 10.48550/arXiv.1801.04381.

[45] Johri, Prashant & Khatri, Sunil Kumar & Al-Taani, Ahmad & Sabharwal, Munish & Suvanov, Shakhzod & Chauhan, Avneesh., “Natural Language Processing: History, Evolution, Application, and Future Work,” 2021. doi: 10.1007/978-981-15-9712-1\_31.

[46] “What Is NLP (Natural Language Processing)? | IBM.” Accessed: May 05, 2025. [Online]. Available: https://www.ibm.com/think/topics/natural-language-processing

[47] “What is a confusion matrix? | IBM.” Accessed: Apr. 22, 2025. [Online]. Available: https://www.ibm.com/think/topics/confusion-matrix

[48] Schlosser, Tobias & Friedrich, Michael & Meyer, Trixy & Kowerko, Danny., “A Consolidated Overview of Evaluation and Performance Metrics for Machine Learning and Computer Vision.,” 2024.

[49] Cutting, Vineesh & Stephen, Nehemiah, “A Review on using Python as a Preferred Programming Language for Beginners,” vol. 8, pp. 4258–4263, 2021.

[50] A. Paszke *et al.*, “PyTorch: An Imperative Style, High-Performance Deep Learning Library,” 2019, *arXiv*. doi: 10.48550/ARXIV.1912.01703.

[51] “What is PyTorch? | IBM.” Accessed: Apr. 23, 2025. [Online]. Available: https://www.ibm.com/think/topics/pytorch

[52] “PyTorch Grows as the Dominant Open Source Framework for AI and ML: 2024 Year in Review,” PyTorch. Accessed: Apr. 23, 2025. [Online]. Available: https://pytorch.org/blog/2024-year-in-review/

[53] B. McFee *et al.*, “librosa: Audio and Music Signal Analysis in Python,” presented at the Python in Science Conference, Austin, Texas, 2015, pp. 18–24. doi: 10.25080/Majora-7b98e3ed-003.